Underspecified parallelism constraint

Frank Schilder and **Markus Guhe**
Department of Informatics
Hamburg University, Germany
{schilder|guhe}@informatik.uni-hamburg.de

Abstract

We propose a modification of the Constraint Language for Lambda Structures (CLLS): an *underspecified* parallelism constraint. The original parallelism constraint offers a descriptive formalisation of simple VP ellipses by indicating two parallel segments of an underspecified tree representation. However, these segments still have to be extracted by hand or to be captured by explicit syntactic constraints (Egg and Erk, 2001) for every given example. The underspecified parallelism constraint overcomes this shortcoming by leaving open the exact size of the parallel segments. Moreover, our extension of CLLS accounts for more data, covering ambiguities that arise from gapping, as in *Peter gave Mary a book and John, too*.

1 Introduction

The Constraint Language for Lambda Structures (CLLS) is a formalism for representing underspecified semantic structures (Egg et al., 2001). Following the semantic accounts to ellipsis (Dalrymple et al., 1991; Gardent and Kohlhase, 1996), CLLS successfully employs a constraint, called the 'parallelism constraint', for the description of simple VP ellipses (e.g. (1)) and the analysis of so-called 'Hirschbühler sentences' (e.g. (2)) (Hirschbühler, 1982).

(1) Mary sleeps and John does, too.

(2) Every computer scientist went to a workshop. Every linguist did too.

With the parallelism constraint it is possible to mark the parallelism of that segment of the first sentence that corresponds to the elliptical segment of the second sentence. For example, in (1) this is the parallelism in the VP between *Mary sleeps* (source sentence) and *John does, too* (target sentence) representing that John sleeps.

In contrast to the semantic accounts that employ higher-order unification, the parallelism constraint directly affects the underspecified semantic structure. Hence, there is no need to keep track of the occurrence of subterms (Gardent and Kohlhase, 1996). The parallelism constraint requires to mark the segment in the semantic representation of the source sentence (e.g. *Mary sleeps*) that is parallel to the segment of the target sentence (e.g. *John does, too*). So far the parallelism constraint in CLLS is not defined in a way that allows to automatically identify the segment in the source sentence. Instead, this segment must be extracted by hand or be derived by syntactic constraints. We propose a general solution to this problem that does not have to rely on a purely syntactic derivation of the source segment as proposed by Egg and Erk (2001).

In short, our solution consists in introducing an *underspecified* source segment that gets specified by the modified parallelism constraint. Additionally, an underspecified source segment does not only cover VP ellipses as in (1) and (2) but also ambiguities that arise from gapping (e.g. (3a)). Consider the following sentences:

(3) a. Peter gave Mary a book and John, too.
b. Peter gave Mary a book and John did, too.

In (3a) gapping allows for two plausible readings:

- *Peter gave Mary a book and Peter gave John a book.*
- *Peter gave Mary a book and John gave Mary a book.*

In addition, a third reading is conceivable, which, however, is ruled out on pragmatic grounds.[1]

[1] The third reading becomes the preferred one in a different context: *Peter gave Mary a book and John, too. John was Peter's favourite doll.*

- *Peter gave Mary a book and Peter gave Mary John.*

In (3b), on the other hand, there is no ambiguity of this kind. Note that the parallelism constraint has to be employed *three* times, each in a different way, to capture the ambiguity expressed by (3a) and to obtain all possible readings. Hence, an underspecified representation for the the three readings of (3a) would be advantageous. Consequently, the proposal made by this paper suggests an *underspecified parallelism constraint* to provide one concise description that covers all conceivable readings.

Our proposal uses the underspecification of the parallelism constraint in order to obtain a single representation for the three readings. The proposed formalisation leaves open the exact size of the source segment. We determine the different readings by filtering out the appropriate segments via the *type* information of the nodes. The nodes in the target segment must have the same type as the corresponding nodes in the source segment.

Up to now, few proposals have been made regarding the automatic determination of source and target sentences in VP ellipses. Hardt (1997) uses different heuristics to rank all VPs within the previous three sentences as potential source VPs. This method covers a larger set of possible source VPs than necessary. Egg and Erk (2001) propose a compositional account of VP ellipses within the HPSG framework. Since they also employ CLLS for the semantic representation of the VP ellipsis their approach is similar to our proposal on several points. On the other hand, their approach is limited to VP ellipses where they provide explicit constraints within the grammar formalism used. In addition, their account does not yet handle gapping such as in (5). In particular, they do not present an *underspecified* representation for this phenomenon.

Clearly, the underspecified parallelism constraint may offer too many readings, as there are three readings possible for (3a) given only the type constraint. The third reading for (3a) is excluded due to a sortal restriction of *to give*, which is not part of the CLLS formalism. We assume that further pragmatic and discourse constraints can be added to the CLLS description excluding pragmatically implausible readings.[2]

The structure of the remainder of the paper is as follows. First, we provide a brief introduction to the formalism CLLS explaining the parallelism constraint in more detail. Then, we present the extension of the parallelism constraint in section 2.2. Finally, we conclude and give an outlook of ongoing work in section 3.

2 Underspecified parallelism

2.1 CLLS and the parallelism constraint

The Constraint Language for Lambda Structures (CLLS) provides a formal framework for the partial description of lambda structures. Lambda structures are tree-like structures representing λ-terms. The lambda term $\mathrm{Mary}(\lambda x.\mathrm{sleep}(x)) \wedge \mathrm{John}(\lambda P.P)$, for example, is represented as the tree structure indicated by figure 1.[3] Formally, lambda structures are trees amended by two partial functions: lam for abstraction and binding variables and ante for modelling anaphoric expressions. The tree possesses further labels and lines indicating application (@), variables (var) as well as the binding relation between the variable and the binder (the dashed arrow). Additionally, variables denoting tree nodes

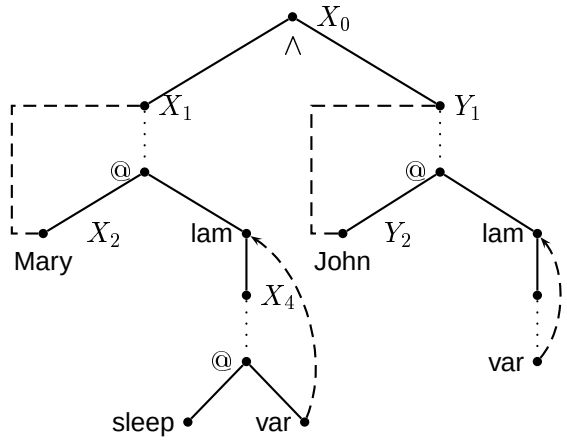


Figure 1: The parallelism constraint for the elliptical sentence in (4)

(e.g. X_0, X_1 in figure 1) and a transitive and reflexive dominance relation ($\lhd^*$) holding between tree nodes allow for underspecification, the specification of anaphoric references etc. A CLLS formula – a

[2]Instead of an underspecified representation for (5), approaches that rely on discourse relations could offer a preferred reading (Kehler, 2000; Asher, 1993). However, explicit world knowledge is required for the formalisation of VP ellipsis and gapping within these frameworks.

[3]Note that all NPs including PNs are type-raised. The term Mary in $\mathrm{Mary}(\lambda x.\mathrm{sleep}(x))$, for example, is a function from sets of entities to truth values.

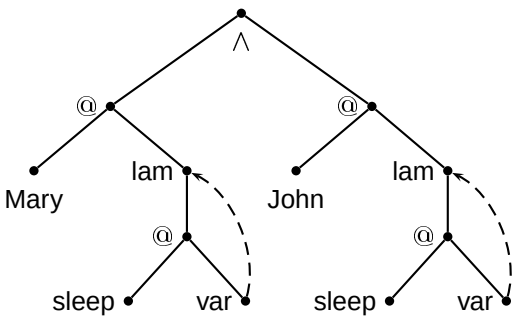


Figure 2: A lambda structure that satisfies the constraint graph in figure 1

conjunction of atomic literals – is a partial description of lambda structures. Every CLLS formula can be graphically represented as a constraint graph, as in figure 1. In order to satisfy this formula a lambda structure and a variable assignment have to be found such that every literal is satisfied. A CLLS constraint φ is defined as follows: $\varphi ::= X : f(X_1, \ldots, X_n) \mid X \lhd^* Y \mid \lambda(X) = Y \mid \mathrm{ante}(X) = Y \mid X_1/X_2 \sim Y_1/Y_2 \mid \varphi \wedge \varphi'$

Since the parallelism constraint $X_1/X_2 \sim Y_1/Y_2$ is crucial for the purpose of this paper and, in addition, more complicated than the others, more details are provided in the following. $A \sim B$ is satisfied iff the segment A of the lambda structure is parallel to a segment B. Segments in a lambda structure are defined as X/Y where X denotes the root of the segment and Y a hole such that $X \lhd^* Y$. The segment covers all nodes that are dominated by the root X with the exception of the nodes dominated by Y. In other words, a segment is a sub-tree starting with the node X with the exception of a further subtree whose root node is Y. In figure 1 the segment X_1/X_2 has the root node X_1 and includes all nodes dominated by X_1 apart from node X_2. Sometimes, the segments are indicated by brackets in the constraint graphs (see figure 1), but normally the constraint is explicitly added to the constraint graph. Formally, the parallelism between two segments is captured via a correspondence function which is defined as a bijective mapping between the nodes of the two segments.[4]

The parallelism constraint has proven to be especially useful for the description of VP ellipses, as in the following example sentence:[5]

(4) Mary sleeps and John does, too.

The CLLS description for the sentence in (4) can be found in figure 1. The parallelism constraint $X_1/X_2 \sim Y_1/Y_2$ is reflected by the graph via two dashed brackets denoting the two parallel segments (i.e. X_1/X_2 and Y_1/Y_2). The brackets precisely determine the part of the source segment that has to be copied into the target segment as well as the part that has to be kept separate (i.e. Mary and John). A lambda structure that satisfies the constraint graph in figure 1 is given in figure 2.

In addition to a descriptive formalisation of VP ellipses CLLS also offers a syntax–semantics interface. Egg et al. (2001) provide a grammar and the rules for the semantic construction of a wide range of example sentences covering phenomena including scope ambiguities and VP ellipses. However, they assume that the parallelism constraint is generated by an independent source, because the determination of the elliptic element is still an open research question. We extend the constraint such that all possible readings for an elliptical sentence can be automatically generated by the semantic construction of the syntax–semantic interface.

2.2 Underspecified parallelism constraint

Elliptic sentences as in (3a) show the need for an underspecified parallelism constraint, since a gapping ambiguity can arise. The formalisation of an underspecified parallelism constraint requires a change of the $X_1/X_2 \sim Y_1/Y_2$ constraint. Instead of naming one segment X_1/X_2, a set of possible segments fulfilling the parallelism constraint is given. This set of segments is captured by an underspecified segment $X/^*Y$ defined as follows. An underspecified segment $X/^*Y$ is the set of all segments that have the root node X and as a hole any node Z for which $Y \lhd^* Z$ holds.

Definition 1 (underspecified segments) *An underspecified segment $\pi/^*\mu$ is a set of segments such that*

$$\pi/^*\mu := \{\pi/\mu' \mid \mu \lhd^* \mu'\}$$

In addition, the set of all nodes in a segment is defined as follows (Egg et al., 2001):

$$\mathsf{b}^-(\pi/\mu) := \{\pi' \mid \pi \lhd^* \pi', \text{but not } \mu \lhd^* \pi'\}$$

[4] See Erk et al. (2001) for a semi-decision procedure and further details on processing complexity.

[5] CLLS can provides an explanation for the following phenomena: the interaction between scope and ellipsis as given for the Hirschbühler sentences and the interaction between anaphora and ellipsis leading to ‘sloppy’ and ‘strict’ readings of anaphoric expressions (Egg et al., 2001).

The definition of an underspecified segment offers too large a set of possible solutions for the parallelism constraint, and most of them are also wrong. Hence, further constraints on the holes for $X \lhd^* Y$ have to be imposed. We propose a filter that selects all the appropriate segments by comparing the type of every node of the source segment with every node of the target segment.

Though CLLS does not come with a type system, the type information can easily be added to the lambda structure definition. In Type Theory a type is defined as follows (Gamut, 1991):

Definition 2 (types) *The set of types* $\mathbf{T}$ *is the smallest set such that:*

1. $e, t \in \mathbf{T}$
2. *if* $a, b \in \mathbf{T}$*, then* $\langle a, b \rangle \in \mathbf{T}$

The two basic types e and t refer to an entity and a truth value, respectively.The predicate $love$, for instance, is of type $\langle e, \langle e, t \rangle \rangle$: a function from two entities to truth values. Note that type-raised representations of NPs are used. Hence, determiners are treated as generalised quantifiers (Barwise and Cooper, 1981) with type $\langle \langle e, t \rangle, \langle e, t \rangle, t \rangle$.

Given a type system for the lambda structures, the correspondence function c must fulfil the following constraint **P5** in addition to the constraints **P1**–**P4** already defined in Egg et al. (2001). The improved definition for the parallelism constraint now covers all 5 constraints and takes into account the definition of an underspecified segment.

Definition 3 (underspecified parallelism) $\pi/^*\mu \sim \pi_2/\mu_2$ *holds in a lambda structure iff there exists a segment* $\pi_1/\mu_1 \in \pi/^*\mu$ *and there is a correspondence function* c *between* π_1/μ_1 *and* π_2/μ_2 *such that* **P1**–**P5** *are satisfied:*

P1–P4 *see Egg et al. (2001)*

P5 *All parallel nodes possess the same type:*

$$\forall \pi \in \mathsf{b}^-(\pi_1/\mu_1) : \mathrm{type}(\pi) = \mathrm{type}(c(\pi))$$

This new definition for the parallelism constraint can handle an underspecified segment for the representation of a gapping ambiguity, such as (3a). It should also be stressed that this new constraint covers simple VP ellipses as well, as in (1). However, we have not yet fully specified how the parallelism constraint can be automatically derived from the syntactic and semantic analysis.

The grammar given by Egg et al. (2001) treats the elliptic *does (too)* as an intransitive verb without adding any further constraints. The semantics contributed by this expression is left open and is to be filled by the parallelism constraint which is computed separately. Our proposal offers a syntax–semantics interface that provides the semantic construction for elliptic sentences. Hence we first add some grammar rules to the CLLS fragment:

(a14) `VP → DTV NP NP`
(a15) `S → S Coord ES`
(a16) `ES → NP Part`
(a17) `ES → NP VPE Part`
(a18) `VPE → does`
(a19) `Part → too`

These new phrase structure rules capture ditransitive verbs `DTV` **(a14)** and elliptic sentences `ES` (**(a15), (a16), (a17)**). Furthermore, words and lexical categories are introduced: elliptic *does/do/did*,[6] (`VPE`) and particle *too* (`PART`). In addition to describing the new rules and categories we will also repeat some of the original rules because they are needed for the example sentences discussed in this paper.[7]

(a1) `S → NP VP`
(a2) `VP → IV`
(a3) `VP → TV NP`
(a7) `NP → PN`
(a8') `NP → Det N`
(a13) $\alpha \rightarrow$ W, if $(W, \alpha) \in Lex$

Given these rules, the sentence *Peter gave Mary a book and John, too* can be analysed. The resulting syntax tree is found in figure 3.

Semantic construction is based on syntactic analysis. For every phrase structure rule there is a semantic construction rule. For each semantic construction rule more constraints are added until a conjunction of constraints is derived for the entire sentence. Table 1 contains the semantic construction rules **(b7)**, **(b8')** and **(b14)**–**(b17)** which correspond to the phrase structure rules **(a7)**, **(a8')** and **(a14)**–**(a17)**.

Generally speaking, CLLS constraints are given for every construction rule. Further variables are in-

[6] Note that tense information is not yet considered.

[7] We use the phrase structure rule **(a8')** instead of **(a8)** `NP → Det` $\overline{\mathrm{N}}$, because $\overline{\mathrm{N}}$ is not needed for the examples discussed in this paper. Lex is a relation that relates words and lexical categories.

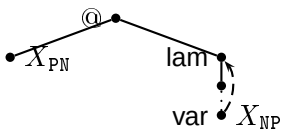
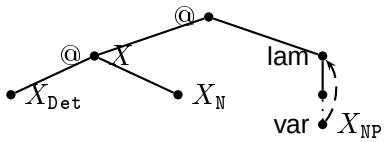
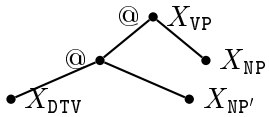
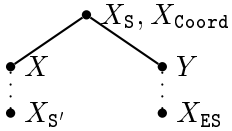
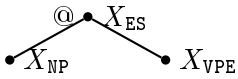
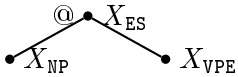
(b7)	$[_{\mathtt{NP}}\ \mathtt{PN}]$	$\Rightarrow$		$X^P_{\mathtt{NP}} = X^P_{\mathtt{PN}}$
(b8')	$[_{\mathtt{NP}}\ \mathtt{Det}\ \mathtt{N}]$	$\Rightarrow$		$X^P_{\mathtt{NP}} = X$
(b14)	$[_{\mathtt{VP}}\ \mathtt{DTV}\ \mathtt{NP}\ \mathtt{NP}']$	$\Rightarrow$		
(b15)	$[_{\mathtt{S}}\ \mathtt{S}'\ \mathtt{Coord}\ \mathtt{ES}]$	$\Rightarrow$		$X/^*X_{\mathtt{S}'} \sim Y/X^P_{\mathtt{ES}}$
(b16)	$[_{\mathtt{ES}}\ \mathtt{NP}\ \mathtt{Part}]$	$\Rightarrow$		$X^P_{\mathtt{ES}} = X^P_{\mathtt{NP}}$
(b17)	$[_{\mathtt{ES}}\ \mathtt{NP}\ \mathtt{VPE}\ \mathtt{Part}]$	$\Rightarrow$		$X^P_{\mathtt{ES}} = X^P_{\mathtt{NP}}$

Table 1: Some semantic construction rules for the syntax–semantics interface

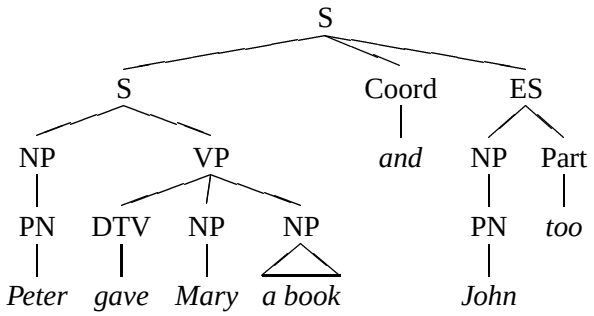
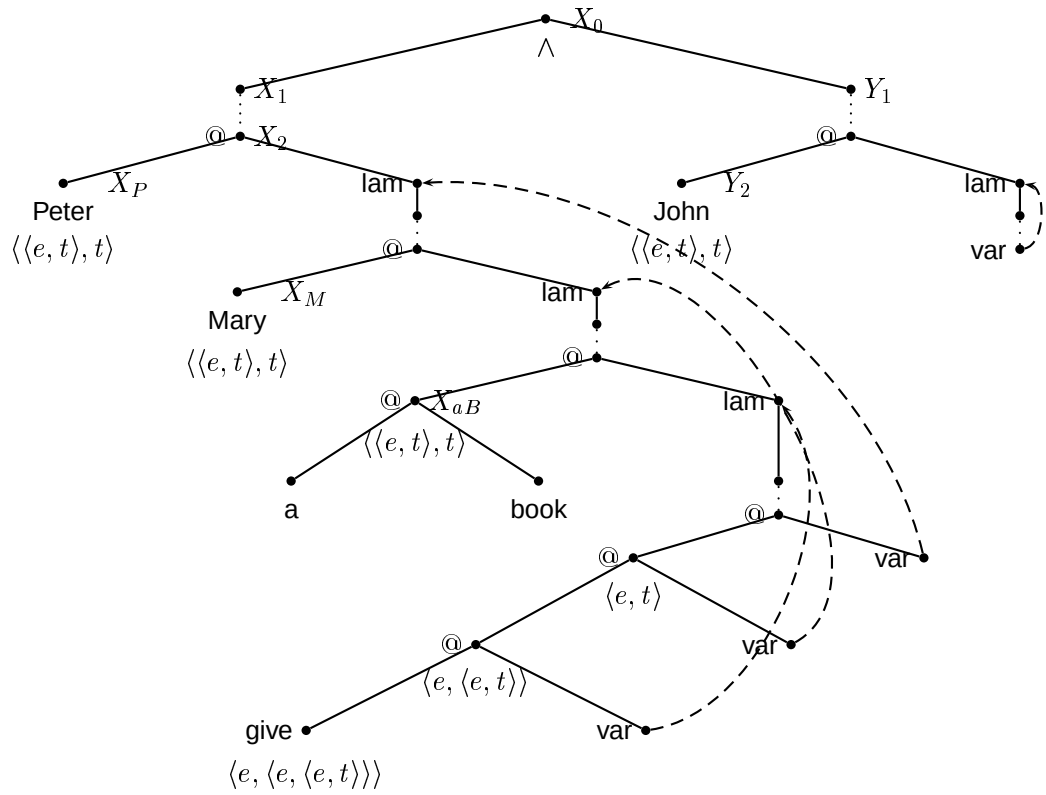


Figure 3: The syntax tree of (3a)

troduced for every syntactic rule: the (root) node labels (e.g. $X_{\mathtt{S}}, X_{\mathtt{NP}}, X_{\mathtt{VP}}$) correspond to the position in the syntactic tree (e.g. $[_{\mathtt{S}}\ \mathtt{NP}\ \mathtt{VP}]$): the label $X_{\mathtt{S}}$ is assigned to category S, $X_{\mathtt{NP}}$ to NP and so forth. That means, further CLLS constraints coming from another phrase structure rule are added wrt. these labels. The construction rule **(b7)** $[_{\mathtt{NP}}\ \mathtt{PN}]$, for instance, uses three labels for the CLLS constraints: $X_{\mathtt{NP}}$ and $X_{\mathtt{PN}}$ as well as $X^P_{\mathtt{NP}}$. The label $X_{\mathtt{NP}}$ refers to the category NP, whereas the label $X_{\mathtt{PN}}$ is connected to the category PN. The label $X^P_{\mathtt{NP}}$ is important for the derivation of the parallelism constraint, similarly to the scope island constraint discussed in Egg et al. (2001).[8] Constraints associated with these labels are added to the respective node in the tree. The nonterminal PN, for example, may be expanded to *John* via the rule **(a13)** PN $\rightarrow$ *John*. In such a case, John is added to the node labelled by $X_{\mathtt{PN}}$.

In our example regarding **(b7)**, the label $X^P_{\mathtt{NP}}$ is equal to $X^P_{\mathtt{PN}}$ which is set equal to the node $X_{\mathtt{PN}}$ labelled by John via **(a13)**. Going up the discourse tree, the rule **(b16)** adds a further equation: $X^P_{\mathtt{ES}} = X^P_{\mathtt{NP}}$. At the next level in the tree the construction rule **(b15)** introduces the parallelism constraint: $X/^*X_{\mathtt{S}'} \sim Y/X^P_{\mathtt{ES}}$. The label $X^P_{\mathtt{ES}}$ refers to the label $X^P_{\mathtt{NP}}$ defined by **(b16)** and this label refers to the

[8] The scope island constraint does not play a role in the examples discussed. Thus, for readability the labels of type X^S are omitted throughout this paper.



$$\wedge \;\; X_1/^*X_2 \sim Y_1/Y_2$$

Figure 4: Underspecified parallelism constraint and type constraints for (5)

node that is labelled by John, as required (cf. figure 4). It should be borne into mind that the equation of the tree nodes (e.g. $X^P_{\mathtt{ES}} = X^P_{\mathtt{NP}}$) makes sure that the correct tree label is inserted into the parallelism constraint. This equation is not a CLLS constraint for describing lambda structures. Instead, the equation is part of the syntax–semantic interface.

Further constraints regarding lexical categories normally only add more labelling constraints such as $X_{\mathtt{PN}} : \mathtt{John}$. Additionally, the type information is introduced at this point (e.g. $\langle\langle e,t\rangle,t\rangle$ for John).

The definition of the semantic construction rules also ensures that the parallelism constraint is introduced. Rule **(b15)** adds this constraint to the representation. Note that at this point of the derivation the target segment is not yet fully specified (i.e. $Y/X^P_{\mathtt{ES}}$). The label $X^P_{\mathtt{ES}}$ depends on the equation constraints that show up for the semantic rule defined for ES and subsequent rules for the NP of the elliptical sentence.

The underspecified source segment (i.e. $X/^*X_{\mathtt{S}'}$), on the other hand, covers all possible elliptical ambiguities (i.e. a subset of $X/^*X_{\mathtt{S}'}$). Together with the parallelism constraint **P5** all conceivable readings can be determined due to the correspondence function. Furthermore, the underspecified parallelism constraint also captures unambiguous VP ellipses better than the previous definition. With the new definition the derivation of the source segment is constrained by the type information that was added to the lambda structure, which gives us the key for the automatic generation of the source segment.

In the following, we provide the derivations of the two example sentences discussed at the beginning of this paper ((5)=(3a) and (6)=(3b)).

(5) Peter gave Mary a book and John, too.

(6) Peter gave Mary a book and John did, too.

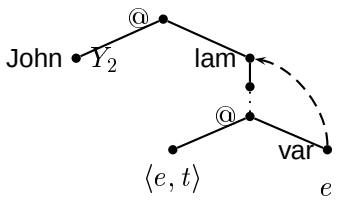


Figure 5: The elliptic sentence *John did, too*

Figure 4 represents the CLLS constraints for sentence (5). Only the types relevant for the parallelism constraint are shown. Three segments can now be specified that fulfil the condition set by the new definitions for an underspecified segment and the underspecified parallelism constraint: X_1/X_P, X_1/X_M, X_1/X_{aB}. All other segments in $X_1/^*X_2$ do not comply with the type equivalence constraint **P5**. Consequently, only three readings are derivable for (5). Sentence (6), on the other hand, blocks two readings that sentence (5) allows. Again constraint **P5** will be applied here. Note that the semantic construction rule for VPE $\rightarrow$ *did* offers the type information $\langle e, t\rangle$ but no further labelling constraint (cf. figure 5). The type information, however, is crucial for the parallelism constraint. Since the type is $\langle e, t\rangle$ the variable var can only correspond to the last argument x in $give(z, y, x)$ which is bound via the λ-operator to the subject of the source sentence. All the other variables have a sister node of a different type (i.e. $\langle e, \langle e, t\rangle\rangle$ and $\langle e, \langle e, \langle e, t\rangle\rangle\rangle$). Hence only one reading is available for (6).

3 Conclusions

In this paper we presented an enhancement of the parallelism constraint defined in CLLS: the *underspecified parallelism constraint*. Besides covering all cases of the original constraint, this new parallelism constraint can handle additional ones, namely ellipses involving gapping. More importantly, we presented the definition of a grammar fragment and semantic construction rules that can automatically extract the target segment(s) of an elliptic sentence. Target segment(s) are generated via the matching of the type information of all nodes of the target segment to the source segment.

Current research focusses on restrictions of possible ambiguous readings due to syntactic reasons, as in 7.[9]

(7) Peter gave a book to Mary, and John, too.

Note that this example is not a counter example to the approach presented because the same mechanism of an underspecified parallelism constraint can be used. Only further constraints have to be specified in order to obtain the correct subset of conceivable readings in $X/^*X_{S'}$.

[9]Thanks to one of the anonymous reviewers, who pointed out this example to us.

References

Nicholas Asher. 1993. *Reference to Abstract Objects in Discourse*, volume 50 of *Studies in Linguistics and Philosophy*. Kluwer Academic Publishers, Dordrecht.

John Barwise and Robin Cooper. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219.

Mary Dalrymple, Stuart Shieber, and Fernando Pereira. 1991. Ellispsis and higher-order unification. *Linguistics & Philosophy*, 14:399–452.

Markus Egg and Katrin Erk. 2001. A compositional account of VP ellipsis. In *8th International Conference on Head-Driven Phrase Structure Grammar*, Trondheim, Norway.

Markus Egg, Alexander Koller, and Joachim Niehren. 2001. The constraint language for lambda structures. *Journal of Logic, Language, and Information*, 10:1–29.

Katrin Erk, Alexander Koller, and Joachim Niehren. 2001. Processing underspecified semantic representations in the constraint language for lambda structures. *Journal of Language and Computation*.

L. T. F. Gamut. 1991. *Logic, Language, and Meaning*, volume 2. The University of Chicago Press, Chicago.

Claire Gardent and Michael Kohlhase. 1996. Higher–order coloured unification and natural language semantics. In *Proceedings of the 34rd Annual Meeting of the ACL*, Santa Cruz. ACL.

Daniel Hardt. 1997. An empirical approach to VP ellipsis. *Computational Linguistics*, 23:525–541.

Paul Hirschbühler. 1982. VP deletion and across-the-board quantifier scope. In J. Pustejovsky and P. Sells, editors, *Proc. of the Twelfth Annual Meeting of the North Eastern Linguistic Society (NELS 12)*, pages 132–139, U. Mass., Amherst.

Andrew Kehler. 2000. Ellipsis. *Linguistics & Philosophy*.