Projecting Propbank Roles onto the CCGbank

Stephen A. Boxwell and Michael White

Department of Linguistics
The Ohio State University
Columbus, OH, USA
boxwell@ling.osu.edu, mwhite@ling.osu.edu

Abstract

This paper describes a method of accurately projecting Propbank roles onto constituents in the CCGbank with near perfect accuracy and automatically annotating verbal categories with the semantic roles of their arguments. The current version of the CCGbank annotates arguments and adjuncts in a suboptimal way – it relies heavily on the Penn Treebank CLR tag, which is widely considered unreliable. By incorporating Propbank roles we are able to modify the derivation to better reflect linguistic reality. Tagging of nodes in the CCG derivation also permits us to annotate verbal categories with semantic roles corresponding to their syntactic arguments, which has strong implications for many NLP tasks.

1. Introduction

The CCGbank (Hockenmaier and Steedman, 2007) is a corpus of derivations in Steedman's (2000) Combinatory Categorial Grammar (CCG) formalism derived from the Wall Street Journal section of the Penn Treebank (Marcus et al., 1993, henceforth PTB). Using a grammar extracted from this resource, Clark & Curran (2007) have recently shown state-of-the-art statistical parsing performance on a cross-parser evaluation based on the syntactic dependencies in the PARC DepBank (King et al., 2003). However, for many NLP applications, one is interested in semantic roles rather than syntactic dependencies, and because the relations between verbs and their arguments can be idiosyncratic, semantic roles cannot always be straightforwardly recovered.

To enable the development of automatic methods of semantic role recovery, the creators of Propbank (Palmer et al., 2005) have annotated the semantic roles of the arguments and modifiers of all verbs in the Penn Treebank. Unfortunately, these annotations cannot be directly transferred to the CCGbank, due to differences in constituent structure between PTB and CCGbank. These discrepancies led Gildea and Hockenmaier (2003) to project Propbank roles onto the CCGbank using head words in their work on semantic role tagging with CCG, with suboptimal results. In this paper, we describe a method of projecting Propbank roles onto constituents in the CCGbank using string alignments, with near perfect accuracy. We also show how these roles can be propagated through the syntactic dependencies in the CCGbank in order to automatically annotate verbal categories with the semantic roles of their arguments. Propagating semantic roles in this way identifies discrepancies between Propbank and CCGbank which do not show up at the string level. Accordingly, we have restructured the derivations in the CCGbank to resolve these discrepancies. Our restructuring has given a small improvement in precision (0.7%) and a larger improvement in recall (3.5%) of core semantic roles. We expect the resulting resource to facilitate research on semantic role labeling and broad coverage generation with CCG.

2. The Propbank

The understanding of how verbs are related to their arguments is central to sentence processing. However, the semantic relations of the arguments can be difficult to derive based on syntax alone, as sentences like (a-c) illustrate:

(a) [Robin]$_{agent}$ opened [the door]$_{theme}$.

(b) [The key]$_{instr}$ opened [the door]$_{theme}$.

(c) [The door]$_{theme}$ opened.

In Propbank, semantic roles have been manually annotated, with the annotations represented in the following way:

(1) open.01 —– 0:1-ARG0 3:0-rel 2:1-ARG1

Each Propbank entry represents a particular instance of a verb in a particular sentence in the treebank. The strings "ARG0", "rel", and "ARG1" represent the semantic roles that certain constituents play in the sentence. Exactly what these roles mean is described on a verb-by-verb basis: in the case of the verb *open*, "ARG0" refers to opener, and "ARG1" refers to the entity or thing being opened, and "rel" represents the verb itself. Another instance of the verb *open* may include ARG3, which refers to the instrument of the opening action. The mapping of numbered roles to precise meanings is given on a verb-by-verb basis in a set of frames files distributed with the Propbank. Note that in addition to these core, numbered roles, Propbank also includes annotations of a variety of modifier roles, prefixed by ARGM.

The numbers in the Propbank entry encode which constituent bears the corresponding semantic relation. The first number represents a terminal that is dominated by the semantic-relation bearing node, and the second number represents the number of levels higher in the tree that the semantic role appears. Although this method is clear and efficient for describing a constituent in the Penn Treebank, problems arise when attempting to locate corresponding constituents in the CCGbank. This is due to a number of discrepancies between the Penn Treebank and the CCGbank. First, the Penn Treebank encodes traces, whereas the CCGbank does not. This creates a mismatch between the

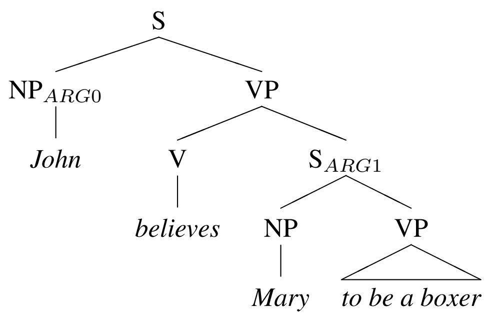

Figure 1: Small clause analysis of *believe* assumed in PTB and Propbank, where *Mary to be a boxer* receives the ARG1 role

terminal indices in PTB and CCGbank. Second, the CCGbank omits some punctuation, like double quotation marks. This further compounds the problem of terminal index mismatches. Third, the CCGbank is constrained by CCG theory to be binary branching, while the Penn Treebank is not constrained in such a way. This creates a mismatch in the number of nodes one must go up a tree to locate the proper constituent for a semantic role. Finally, the Penn Treebank and CCGbank sometimes disagree as to which substrings in the sentence form a constituent. Consider the case of object-control verbs like *believe*: in PTB and Propbank, such verbs take small clause complements that receive an ARG1 role, as shown in Figure 1. By contrast, in the CCGbank, obect-control verbs take NP and VP complements, as shown in the derivation in Figure 2. Consequently, there is no constituent that covers the same substring that ARG1 was intended to cover. This will cause role projection to fail if we only apply rules based on a single constituent covering a given substring. All together, these three differences make it impossible to use Propbank entries directly with the CCGbank.

3. Mapping Roles to CCGBank

These mismatches between the two corpora makes applying Propbank data to the CCGbank a challenge. However, it is possible to resolve these inconsistencies and generate a version of the CCGbank that tags nodes in the derivation with the correct Propbank role. The first step is to find a mapping of PTB terminal indices to CCGbank terminal indices. This allows us to correct for the terminal index mismatch that comes about as a result of missing traces and punctuation. If a particular Penn Treebank terminal does not have a corresponding terminal in the CCGbank, it will be mapped to null and will be ignored by the algorithm.[1] This mapping is easily obtained by listing the terminals of a particular sentence from both the Penn Treebank and the CCGbank and comparing them using the UNIX *diff* utility. Once we have a mapping from Penn Treebank terminals to CCGbank terminals, we identify the node in the Penn Treebank that corresponds to the semantic role that we wish to annotate in CCGbank. Then, we identify the set of terminals that are dominated by that node, and map this set to a set of corresponding CCGbank terminals. We then try to find a constituent in the CCGbank that dominates that set of terminals. If we find a terminal that dominates all and only the terminals in the set, then we have successfully found the correct node to tag with the Propbank role. In cases where we identify a node with no sisters (for example, a single word NP, which is generated from an N category by a unary rule), we assign the role to the highest possible node (in this case, the NP). In most cases (about 97%), we were able to identify a constituent that dominates exactly the same set of terminals as the semantic role in the Penn Treebank.
Nearly all the remaining cases (3%) followed the pattern seen with *believe* in Figure 2. In these cases, we split the argument in two in order to accurately reflect the substring that is covered by the Propbank role (Figure 3). Note that this move is tantamount to redefining the frames files for verbs like *believe* to better match CCGbank; in tagging applications, we expect the mapping to be reversible in case one prefers output that is consistent with the original Propbank frames files. With the *believe* structure resolved in this way, the accuracy rates of perfect constituent matches in the entire CCGbank are nearly perfect, as shown in Table 1.

Arg0	100% (81416/81416)
Arg1	99.925% (114555/114640)
Arg2	99.984% (25075/25079)
Arg3	100% (4215/4215)
Arg4	99.969% (3271/3272)
Arg5	100% (84/84)
ArgA	100% (18/18)
ArgM	99.993% (75195/75200)
Overall	99.977% (419183/419278)

Table 1: Mapping Propbank-labeled constituents from PTB to CCGB

4. Resolving Argument-Adjunct Discrepancies

Once the CCGbank has been accurately annotated with Propbank roles, it becomes possible to improve the CCGbank's syntactic derivations using the semantic information encoded in Propbank. One of the known areas for improvement in the CCGbank lies in the distinction between arguments and adjuncts. Because PTB does not draw a sharp distinction between arguments and adjuncts, CCGbank is forced to rely on the "closely related" (CLR) function tag, which is not consistently annotated. Gildea and Hockenmaier (2003) suggest that the inconsistent treatment of arguments and adjuncts in the CCGbank contributes to suboptimal results on semantic role prediction. However, using the semantic information projected on the CCGbank, we can identify cases where the judgement of the Propbank annotators and the CCGbank disagree. In some cases, a syntactic adjunct is annotated as a semantic argument, as in Figure 4. In other cases, a syntactic argument is annotated as a semantic adjunct, as in Figure 5.

[1]This will cause any semantic role that is not coreferential with a visible constituent to be eliminated in the CCG representation. The information from roles annotated on such traces may be used in future research for the projection of anaphoric semantic dependencies.

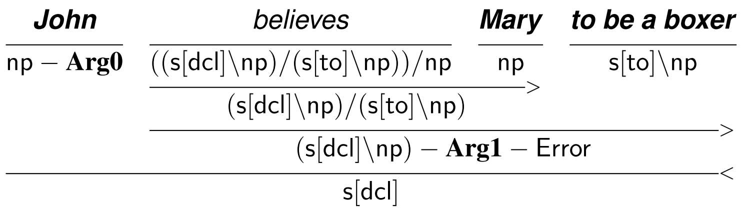

Figure 2: Propbank role does not line up with the constituent structure of CCGbank

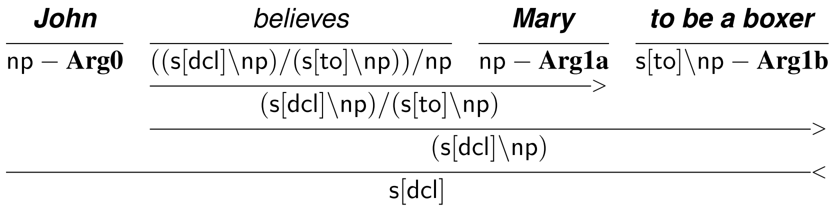

Figure 3: Resolution of the split argument problem

Since the argument/adjunct distinctions in Propbank reflect the carefully considered judgments of the Propbank annotators, it makes sense to modify the derivations in CCGbank to reflect these relationships between verbs and their arguments and modifiers. Doing so should also create more reliable data for machine learning approaches to semantic role prediction. We focused on constituents with the VP-adjunct category (s\np)\(s\np) that were annotated with a numbered Propbank role and constituents with the PP category that were annotated with an ArgM role. We identified 11,569 adjuncts to convert to arguments, and 1543 arguments to convert to adjuncts. We then modified the categories to reflect the judgement of the propbank annotators, making sure to alter the categories below the modified ones (an adjunct-to-argument operation will result in a new PP argument on the verbal category, for example). Finally, we ran a guided parser over the results using the standard CCG combinatory rules to verify that the derivations were not broken in any way by the conversion.

5. Assigning Semantic Roles to Verb Categories

The next step in producing a thoroughly integrated semantic annotation is to propagate semantic role information onto the verbs themselves. Since CCG has the correct domain of locality to directly express predicate-argument dependencies in lexical categories, marking the verbal categories in this way will therefore directly link the syntactic dependencies to the semantic ones, as has always been the intention with Propbank (Babko-Malaya et al., 2006).

To illustrate, consider the example in Figure 6. Notice that the nodes bearing the semantic arguments are the same ones that are consumed by the verb. With this in mind, we can devise a procedure for projecting the semantic roles of the NPs onto the verbal category itself, as in Figure 7. That is, we can use the roles tagged on these nodes to mark up the verbal category as follows:

(2) give.01:$((S[dcl]\backslash NP_{Arg0})/NP_{Arg1})/NP_{Arg2}$

This means that the subject NP should be tagged with Arg0, the first (leftmost) object should be tagged with Arg2, and the second object should be tagged with Arg1. In order to identify which constituent goes with which semantic role, we use the predicate-argument (PARG) files which are automatically generated from the derivations and distributed with the CCGbank.[2] The predicate-argument files delineate the syntactic dependencies of each predicate by giving the index of each argument's head word (see Figure 8).

We can locate the constituent that the verb consumes by generating two paths to the root node: one from the argument's head word, the other from the verb. The *apex* node is the lowest node these two paths share. We then look down the path from the apex node toward the argument head looking for a Propbank role associated with this verb. If we find one, then we can annotate that argument of the verb category with the Propbank role of its associated constituent. Note that this method of identifying the semantic roles of arguments to verbs successfully propagates up the tree through modifiers, relative clauses and other non-local dependencies, as illustrated in Figure 9.

6. Evaluation and Error Analysis

To evaluate the effectiveness of our method of projecting semantic roles onto verbs, as well as to determine the improvement the argument-adjunct modifications make, we measure the extent to which the verbs in the corpus successfully identify their semantic arguments using syntactic dependencies. In particular, we define precision as the percentage of verbal syntactic arguments that are successfully united with their semantic role, and we define recall as the percentage of semantic arguments that locate their verb.[3] The results are shown in Table 2. The argument-adjunct

[2]Note that when annotating the verb in our modified corpus, it is necessary to generate new predicate-argument files from the modified derivations, as some of the word-word dependencies will change along with argument-adjunct relationships.

[3]Note that semantically null NPs (specifically NP[thr] and NP[expl]) are not counted towards the total in the precision measure.

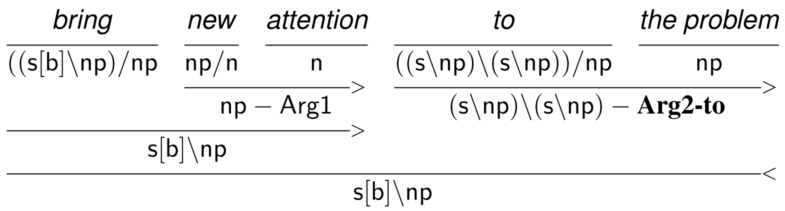

Figure 4: A syntactic adjunct that should be an argument (wsj_0003.4)

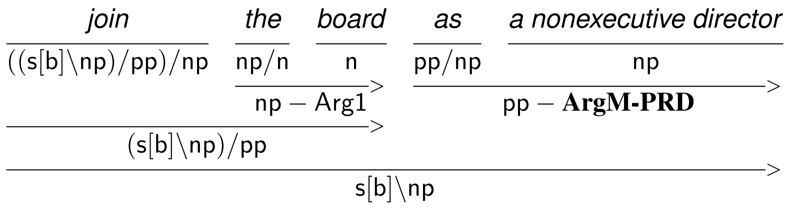

Figure 5: A syntactic argument that should be an adjunct (wsj_0001.1)

CCGbank	Precision	Recall	F score
Original	96.13%	85.71%	90.62%
Restructured	96.85%	89.24%	92.89%

Table 2: Results of semantic-syntactic argument matching (numbered arguments only)

modifications yield a small improvement in precision, due to the removal of the offending ArgM semantic arguments that were annotatated as syntactic arguments in the original CCGbank. The much larger improvement in recall is due to the much larger number of cases of core arguments being annotated as syntactic adjuncts.

Several kinds of errors can occur when projecting semantic roles onto verbal categories. One common case is with *to*-infinitivals where the implicit subject is not actually discharged from the verb — rather, the verb phrase itself is subcategorized for by some other functor or changed by a unary rule, as in Figure 10, and the dependency is not passed on. In this case, *to* dutifully passes the dependency along (like *who* in Figure 9), but the unary rule does not. Consequently, there is no relationship reflected in the CCGbank PARG files, and the verbal category is therefore unable to locate the semantic argument that syntactic argument is associated with, and gives up.

Another common error deals with nominalized control verbs, like the one in Figure 11. Unlike the previous example, in this case the syntactic dependency is passed along all the way to the syntactic argument. Once the verb locates its syntactic argument (*decision*), however, it finds that it is not tagged with a semantic role; the role it is interested in is actually one step up in the derivation. This is not surprising since the implicit subject of *link* should not actually be *decision*, as the CCGbank has it, as can be seen by comparing this sentence to the non-nominalized sentence *Cray Research$_i$ decided t_i to link its ...*. The question of semantic vs. syntactic determination of control is considered at length by Jackendoff and Culicover (2003), who observe that in cases like these the controller is always the relevant agent, which can be expressed in a variety of syntactic positions.

Nominalized control verbs are an example of cases in which semantic roles might best be resolved by means other than syntactic dependencies. Gildea & Hockenmaier (2003) suggest that they should be taken to involve anaphoric dependencies, as there is no syntactic connection between the head words in the CCGbank. Gildea & Hockenmaier also offer the following case where Propbank annotates a semantic relationship between *adhesive* and *apply*:

(3) When properly applied, the adhesive$_{Arg1}$ is designed to... (wsj_0451.5)

Even though *applied* and *the adhesive* are adjacent, there is no syntactic relation between them, as the syntactic relation is between the *when*-clause and the main clause.

It appears that inconsistencies in the treatment of dependencies like these contribute almost all of the semantically null syntactic arguments. To project semantic roles in such cases, we plan to investigate using Propbank roles that are annotated to null constituents (i.e. traces) in the original PTB, which are ignored in CCGbank. Orphaned roles are also likely candidates for resolution to constituents elsewhere in the sentence.

7. Conclusions and Future Work

We expect that our reliable mapping of Propbank roles to CCGbank nodes and the subsequent projection of these roles onto the lexical categories of verbs will prove to be a valuable resource for research on semantic role labeling and broad coverage generation with CCG. Given that CCG's extended domain of locality enables a direct mapping between syntactic and semantic dependencies, we also expect this resource to facilitate research on the joint determination of syntactic and semantic dependencies, where parsing and semantic role labeling are fully integrated. Additionally, the restructuring of arguments and adjuncts in order to be consistent with Propbank roles may lead to improvements in CCG supertagging and parsing as stand-alone tasks.

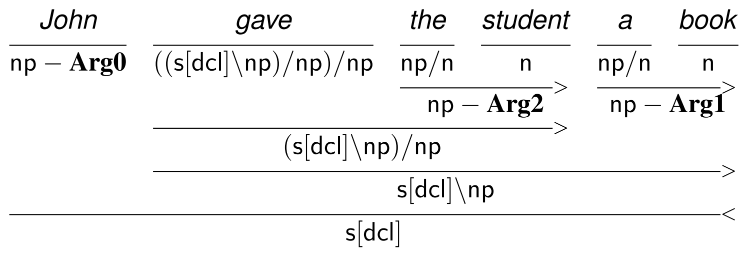

Figure 6: A simple sentence with semantic arguments labeled on constituents

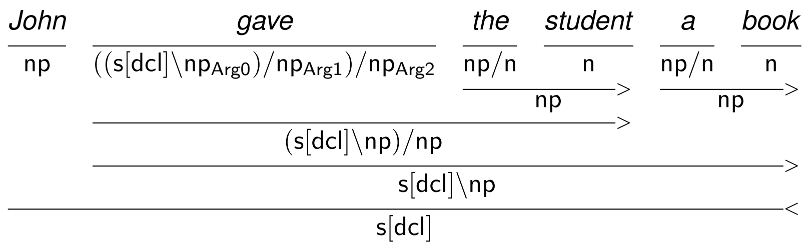

Figure 7: The same sentence with semantic arguments labeled on the verb category

The resource will be made available either in the form of an open source software tool which will generate the modified corpus from the originals, or possibly in a later release of CCGbank. In future work, we plan to address the issue of projecting ArgM roles onto the lexical categories of modifiers. We also plan to refine our approach to handling roles determined anaphorically, annotating the roles which cannot be resolved syntactically in the CCGbank. By doing so, we expect to enable research on employing anaphora resolution techniques as part of a comprehensive approach to semantic role prediction.

Acknowledgements

We would like to thank Julia Hockenmaier for helpful discussion and for making her predicate-argument tool available. We would also like to thank Chris Brew, Detmar Meurers, Eric Fosler-Lussier, and Dennis Mehay for helpful comments.

8. References

Olga Babko-Malaya, Ann Bies, Ann Taylor, Szuting Yi, Martha Palmer, Mitch Marcus, Seth Kulick, and Libin Shen. 2006. Issues in synchronizing the English treebank and propbank. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, pages 70–77, Sydney, Australia, July. Association for Computational Linguistics.

Stephen Clark and James R. Curran. 2007. Formalism-independent parser evaluation with CCG and DepBank. In *Proc. ACL-07*.

Daniel Gildea and Julia Hockenmaier. 2003. Identifying semantic roles using Combinatory Categorial Grammar. In *Proc. EMNLP-03*.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Ray Jackendoff and Peter Culicover. 2003. The Semantic Basis of Control in English. *Language*.

Tracy King, Richard Crouch, Stefan Riezler, Mary Dalymple, and Ronald Kaplan. 2003. The PARC 700 Dependency Bank. In *Proc. 4th International Workshop on Linguistically Interpreted Corpora*.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

Category	Argument Number	Argument Head	Functor Head
(S[dcl]\NP)/PP	1	men_2	worked_4
(S[dcl]\\NP)/**PP**	2	with_2	worked_4
(S\\NP)\**(S\\NP)**	2	worked_4	closely_5
(NP\\NP)/**(S[dcl]\\NP)**	2	worked_4	who_3

Figure 8: Predicate-argument file fragment for wsj_0003.13

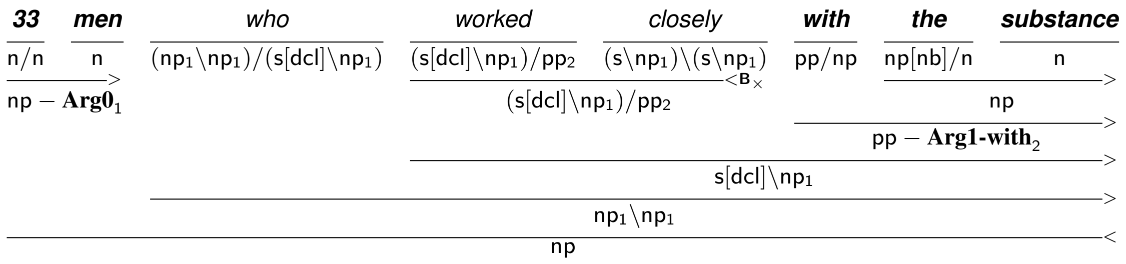

Figure 9: Argument roles can propagate through relative clauses and adverbs (wsj_0003.13)

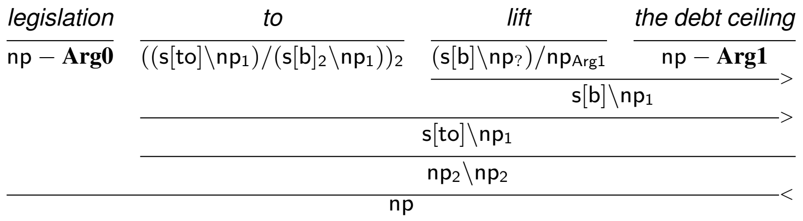

Figure 10: Unary rules can (mistakenly) prevent propagation of verbal dependencies (wsj_0008.4)

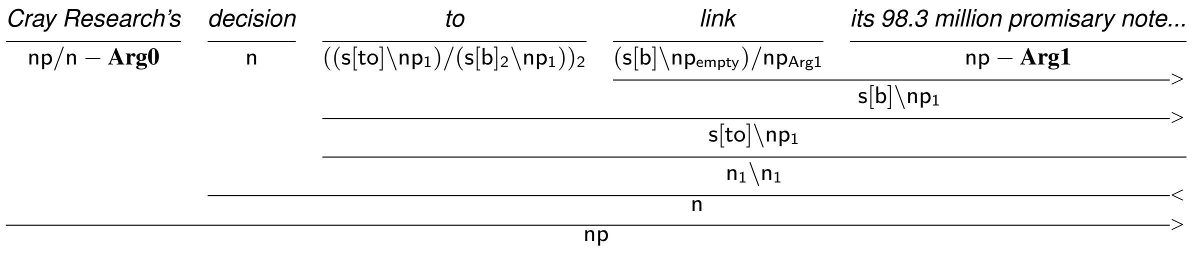

Figure 11: The syntactically inaccessible constituent *Cray Research's* is annotated with the semantic role, rather than the (erroneous) syntactic argument *decision* (wsj_0018.13)