Valency in Casting Systems

Danny Kersten & Gerrit van der Hoeven
University of Twente
Department of Computer Science, Section SETI
PO. Box 217, 7500 AE Enschede, The Netherlands
e-mail: {kersten,vdhoeven}@cs.utwente.nl

Abstract

Casting systems can be used to describe (fragments of) natural language. It is a formalism which associates trees with sentences in the style of classical dependency analysis, see Tesnière (1953). Casting systems in their original form are less elaborate in their linguistic structure, but more formal in the mathematical sense than dependency grammar. This paper describes the notion of casting systems in some detail. The main topic the paper deals with, is the problem of adding the linguistic notion valency to casting systems. Without valency, a casting system will often find an unacceptable amount of different analyses for a given sentence. A lot of these are obviously incorrect or unwanted. Valency can be used to filter out such superfluous analyses.

1 Introduction

This paper is about formal systems which associate sequences of symbols (sentences) with trees. First we will describe an experiment which gave rise to the formalisms we introduce here. The experiment is as follows.

A group of people (a panel) is given a set of sentences which come from a dialog for flight information. As a first step, they are asked to give a structural analysis of the sentences. More precisely: their task is to draw lines between related words in each of the sentences of the corpus, in such a way that every sentence gets a tree structure.

The words of the sentence are the nodes of the tree, the lines connecting mothers and daughters in the tree stand for: 'in some sense related'. Such an analysis applied to the sentence "*The train for London departs from platform one*" might yield a tree like the one in figure 1. There is one restriction concerning word order the panel must obey in drawing the trees: every (sub)tree should describe a contiguous part of the sentence (inorder traversal of the tree yields the sentence).

The second step is, to ask the panel to 'motivate' their tree constructions. The motivation must take a specific form. They are asked to give a name to

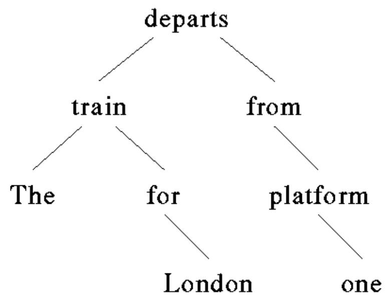

Figure 1

the lines connecting mother- and daughter-words in their trees, in a consistent way for all sentences. In this way they are supposed to make explicit which relations between words they consider important. Moreover, they are asked to name the characteristics of the individual words in every tree. Thus it is made explicit what the properties of the individual words are that make them fit in a particular relationship to one another.

The final step would be, to redraw the trees in such a way that the relation names assigned to lines connecting mother and daughter, are now assigned to the daughter, together with the characteristics of the daughter-word. From this second representation, the original one can be easily reconstructed, since there is always only one line in the tree to which the relation component in the dressing of a daughter can belong. This final tree-translation does not affect the structure of the trees nor does it contribute to the insight into their structure. It is relevant for technical purposes: now we have trees in which only the nodes have attributes, instead of both nodes and arcs.

A schematic representation of a final result tree is shown in figure 2, here we used three different attributes (SL, NL and S for respectively 'Sentence & Leading', 'Noun & Leading' and 'Supporting'). These (syntactical) roles are introduced just for this particular example. In real life we will mostly use semantic attributes when we use this system to describe (part of) natural language.

The resulting description of the sentences could be interesting for all sorts of reasons. Our interest in the outcome is simply to use the trees, the word characteristics and the relations between words as indicated by the panel, to construct similar trees for sentences that they did not consider (probably within the same domain).

The basic idea for 'extrapolation' of the results of the experiment is to abstract from the trees that are delivered, and to concentrate on 'word profiles' that can be derived from the trees. A word profile is roughly a triple consisting of an attribute, and two sets of attributes. A profile for a word can be derived from a set of trees by first collecting all trees in which the word has the same attribute, then collecting the set of attributes assigned to daughters of that

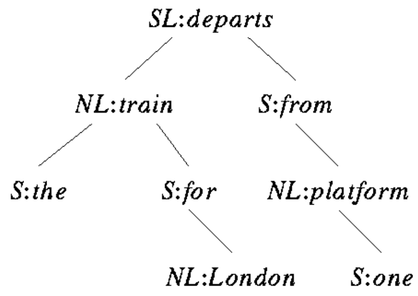

Figure 2

word in any of the trees, and finally splitting this set in two, possibly overlapping, subsets, consisting of attributes assigned to daughters which occur to the left of the given word and the attributes assigned to words which occur to the right. Note that a word can occur with different attributes, and that therefore a word can have more than one profile. The formal notion of a casting system introduced below, gives the precise elaboration of this idea.

The word profile does not restrict the number of possible arcs leaving from a node. This means, if we look at the example once more, that to the left of '*SL:departs*' we can find more than one leading noun. This may be an advantage as well as a disadvantage, which will be addressed later.

Before we turn to the formal definitions, we should take a closer look at the kind of trees we consider and the notion of 'word profile'. In shape, the trees are very much like dependency trees. What we ask our panel to do, could rightly be called dependency analysis. The syntactical claims in our approach however, are far from classical 'dependency syntax'. In fact, we will present a system that is capable of assigning trees to well-formed utterances, but that will assign trees just as easily to many ill-formed utterances. The question what makes a sentence or phrase 'correct', let alone the explanation of correctness at any level of adequacy, does not interest us. What we want, is to have a tree shaped representation of an utterance which organizes the information in that utterance in a way that is both manageable and acceptable to a human reader or hearer of the utterance. As for the word profiles, if one thinks of the attributes for words as semantic categories, and omits the left-of / right-of distinction, a word profile bears some resemblance to a case frame. In fact, it seems that the analysis we consider here could just as well be performed on the basis of a dictionary of case frames, as on the basis of a dictionary of word profiles that are derived from a corpus of hand made analyses.

2 What are Casting Systems?

A casting system is nothing but the formal description of a dictionary of word profiles, as introduced informally above. There is a slight change of terminology however. What we called 'words' above, are 'actors' in the formal representation, and what we called 'attributes', are now 'roles'. We view a sentence as a play; the casting system tells which actors can play which roles, and what supporting roles the actors in their roles expect to their left and to their right.

Strictly formal, a *casting system* is a seven tuple of sets, symbols and relations. It fixes a relation between sequences of actors and dependency trees. It is a dictionary of words, word-roles, and co-occurrence relations between words and roles.

In our example we can recognize actors (*the, train, ...*) and roles (*SL, NL, S*). A node of the tree describes which role can be played by an actor (the role SL can be played by the actor depart, role NL can be played by the actors *train, platform* and *London, ...*). The root of the tree is a special node, this node can only be filled with specific roles (here the leading role is SL). Last the arcs of the tree describe which roles we can find to the left (or right) of a node (to the left of depart in the role SL we can find a role NL, to the right of platform in the role NL we can find a role $S, \ldots$). This leads to the following formal description of a casting system.

Definition
A *casting system* **S** is a seven tuple with the following components:

- $\boldsymbol{A}$, the *actor* set of **S**. $\boldsymbol{A}$ is a finite alphabet. Its elements are *actors*.
- $\boldsymbol{P}$, the set of *roles* of **S**. $\boldsymbol{P}$ is a finite set.
- $\boldsymbol{L}$, the set of leading roles of **S**. $\boldsymbol{L}$ is a subset of $\boldsymbol{P}$.
- ι, the *invisible role* of **S**. ι is a distinguished element of $\boldsymbol{P}$.
- $- : -$, the '*can-be-played-by' relation* of **S**. It relates roles and actors. If P is a role and a is an actor then we write $P : a$ to express that P can be played by a.
- $- \backslash - : -$, the '*can-be-combined-left' relation* of **S**. It relates roles with actors and roles. If P and Q are roles, and a is an actor, then we write $Q \backslash P : a$ to express that a in role P can play together with any actor in role Q to its left.
- $- : - / -$, the '*can-be-combined-right' relation* of **S**. The counterpart of the previous relation in the following sense: we write $P : a/Q$ to express that a in role P can play together with any actor in role Q to its right.

It should be obvious, that the can-be-played-by, can-be-combined-left and can-be-combined-right relations give us the ingredients of a word profile. The special status of the set of leading roles is, that it contains the roles that can appear at the root of a well-formed tree. The invisible role is important in the 'combine' relations. Possibility of combination with the invisible role indicates that an actor can occur without support of other roles, i.e. without daughters in a dependency tree.

If role P can be played by actor a, then the can-be-combined-left and -right relations must be specified for this combination. Or:
$P : a \leftrightarrow \exists Q \in \boldsymbol{P} \mid Q\backslash P : a$
and
$P : a \leftrightarrow \exists R \in \boldsymbol{P} \mid P : a/R$

A casting system is just the rules of the game. The rules can be derived from a given set of trees. But the game is the inverse: to associate trees with sequences of actors. That is what the following definition is about.

Definition
Let $\mathbf{S}$ be a casting system with actor set $\boldsymbol{A}$, and let $\underline{u}$ be a string of actors. A *casting tree* or a *dependency tree* for $\underline{u}$ w.r.t. $\mathbf{S}$ is a directed graph $\underline{T}$. The nodes of $\underline{T}$ are pairs (P, α), with P a role of $\mathbf{S}$, and α an occurrence of an actor of $\mathbf{S}$ in $\underline{u}$. The graph $\underline{T}$ has the following properties:

- it is a *tree*;
- the role of every node can be played by the actor of the node;
- if (Q, β) is a successor of (P, α) and the occurrence β is to the left of the occurrence α, then $Q\backslash P : a$, where a is the actor of which α is an occurrence; if β is to the right of α, then $P : a/Q$;
- if node (P, α) has no successors (Q, β) with β to the left of α, then $\iota\backslash P : a$; if there are no successors (Q, β) with β to the right of α, then $P : a/\iota$;
- with every node there is a segment $\underline{v}$ of $\underline{u}$, which consists of the actors in the node and in its descendants. In particular, the root-node corresponds to the entire sequence $\underline{u}$.

The existence of dependency trees w.r.t. a given casting system determines the strings of actors in a formal language.

Here we will give the full description of the casting system which corresponds to the tree of figure 2.

Actors	*the* *train* *for* *London*	*departs* *from* *platform* *one*
Roles	SL NL	S ι
Leading role	SL	
Invisible role	ι	
'Can-be-played-by'	$S : the$ $NL : train$ $S : for$ $NL : London$	$SL : departs$ $S : from$ $NL : platform$ $S : one$
'Can-be-combined-left'	$\iota \setminus S : the$ $S \setminus NL : train$ $\iota \setminus S : for$ $\iota \setminus NL : London$	$NL \setminus SL : departs$ $\iota \setminus S : from$ $\iota \setminus NL : platform$ $\iota \setminus S : one$
'Can-be-combined-right'	$S : the \,/\, \iota$ $NL : train \,/\, S$ $S : for \,/\, NL$ $NL : London \,/\, \iota$	$SL : departs \,/\, S$ $S : from \,/\, NL$ $NL : platform \,/\, S$ $S : one \,/\, \iota$

In this description, both actors and roles are written in italics. The roles start with a uppercase character, where the actors start with a lowercase character (except for the actor *London*).

3 (Short) Description of Valency

Before we shall combine the casting systems, described in the previous section, with the (linguistic) notion of valency, we look at the notion valency itself.

Valency is defined in many different ways, as is pointed out by Schubert (1987), pp. 61–62. The following example will show what can be expressed by valency. The example is based on the description of valency which can be found in Appel (1992). If a sentence does not meet the requirements of valency, then this sentence is said to be (grammatically) incorrect.

Example:
If we look at the verb *to chop down*, then we know that in a sentence we may expect a *person* to the left of this verb and a *tree* to the right. The person is the acting agent (subject), the tree is the object on which the action is performed (objective). Let us take a closer look at the following sentences:

1. The man chopped the tree down.
2. She chopped down.
3. He chopped a willow an oak down.

4. He chopped a building down.

The person and the tree (as can be found in the first sentence) are the arguments of the verb. We also know that every argument has to be present exactly once for the sentence to be correct. Therefore the first sentence meets the 'valency requirements'. In sentences 2 and 3 the arguments are of the correct type, but the number of arguments is not correct. The last sentence is not correct because the second argument should be a tree (or a tree-like object).

Note:
Some verbs can have optional arguments as well. The examples in section 4 show that the verbs to see and to sell have optional arguments to the right.

The 'requirements' described in the example are part of the notion valency. To be precise, valency in general can describe the following aspects:

Number:	how many arguments we may expect to the left and the right of a word (arguments can be optional);
Order:	the order of the arguments;
Type:	the kind of arguments;
Function:	The function of the arguments w.r.t. the word.

Valency as described by Tesnière (1953) only covers the **number** and **type** of arguments which can be found to the left and the right of a *verb*, and the order of the arguments is not bound. Appel (1992) defines valency with the aspects **number**, **type** and **function** for every lexical element. Note though that in most languages the order of the arguments follows more or less from the function which the arguments perform.

Here we will use valency in a very general way, where *all* words can have a limited number of arguments. The roles of the casting system can be used to describe the type and (part of) the function of the arguments. We will show which aspects of valency can be incorporated in a casting system and we will show some consequences.

4 How Can We Add Valency to Casting Systems?

The casting systems as defined in section 2 incorporate part of the notion of valency. In a casting system one can express that an actor (playing a role) can be combined with other actors of a specific kind; this part can be used as a combination of the *type*- and the *function* aspect of valency described in section 3. We can not specify with how many actors the actor can be combined, nor whether combinations are obligatory or not. This part, the *number aspect*, is not incorporated in casting systems, although it could be useful. Examples of the relevance of the number aspect are shown below. Finally, the *order aspect* of valency can be found in casting systems, but only partially. A casting system does allow the specification of the position of an argument w.r.t. a word, i.e.

it tells which arguments we can find to the *left* and which arguments to the *right*. In a casting system we cannot specify the position of arguments w.r.t. one another. E.g. we can not specify that role A should be to the left of role B if role A and role B both occur to the right of an actor-role combination. We shall not further consider the order aspect of valency in this paper. We do consider the number aspect in some detail. The following sections show how we can enrich casting systems with this aspect.

4.1 Ways to Add the Number Aspect of Valency to a Casting System

Here we will show how to add the number aspect to casting systems. To be precise, we will make some modifications to the definition of a casting system, which will allow us to restrict the admissible dependency trees. There are essentially two modifications that we shall discuss. Neither of the two strong enough to capture fully the number aspect of valency. The combination of the two modifications is probably the best solution.

Before we show the modifications, we will first look at some illustrative examples, using the verbs *see* and *sell*.

To present the examples we first need a limited vocabulary of actors and roles. The words (i.e. actors) we shall use are *furniture, her, I, see, sell, that.* For the roles we shall use an alphabet which is not completely realistic. But now the roles are closer to what may be expected in a 'real-life' casting system than they were before: as roles we use parts of *feature structures.* The roles we consider in the example are: (Subj, Nominative, Animate), (DObj, Accusative, Animate), (DObj, Accusative, Inanimate), (Iobj, Dative, Animate), (Act, Transitive), (Det, Demonstrative), (Det, Possessive). In our example casting systems we shall find can-be-played-by relations with combinations of roles and actors like:

(Subj, Nominative, Animate) : *I*	(Act, Transitive) : *see*
(DObj, Accusative, Inanimate) : *furniture*	(Act, Transitive) : *sell*
(DObj, Accusative, Animate) : *her*	(Det, Demonstrative) : *that*
(IObj, Dative, Animate) : *her*	(DObj, Accusative, Inanimate) : *that*
(Det, Possessive) : *he*r	

What sentences can be made with the word *see* within our limited alphabet (actor set)? The following sentences are possible:

- I see.
- I see her.
- I see furniture.
- I see her furniture.

In these sentences, *see* is the head of the sentence. To the left of *see* we find the subject, in a nominative form with a feature animate. To the right of *see* we find either nothing, or a direct object, in an accusative form, and with at least two possibilities for other features: animate and inanimate. A casting system $\mathbf{S_1}$ which associates appropriate dependency trees with these sentences can be defined as in the next table. In presenting the table we use some almost self-explanatory abbreviations: e.g. SubNomA is (Subj, Nominative, Animate), ActT is (Act, Transitive) and DobAccI is (Dobj, Accusative, Inanimate).

Casting system $\mathbf{S_1}$	
$SubNomA \setminus ActT : see$	$ActT : see\ /\ \iota$
	$ActT : see\ /\ DobAccI$
	$ActT : see\ /\ DobAccA$
$\iota \setminus SubNomA : I$	$SubNomA : I\ /\ \iota$
$\iota \setminus DetPos : her$	$DetPos : her\ /\ \iota$
$\iota \setminus DobAccA : her$	$DobAccA : her\ /\ \iota$
$\iota \setminus DobAccI : furniture$	$DobAccI : furniture\ /\ \iota$
$DetPos \setminus DobAccI : furniture$	

Note that the casting system $\mathbf{S_1}$ assigns trees to ill-formed sentences like the following as well:

- I see her her.
- I I I see.

But working, as we do, on the presupposition that a casting system should be capable of assigning good trees to correct sentences, rather than distinguish between correct sentences and non-sentences, this is not really a problem. Far worse is the following. The correct sentence *I see her furniture* has two dependency trees. One of them is appropriate. The other one is simply incorrect. They are shown in figure 3. The solution of this problem could be a mechanism which forces the verb *see* in its *ActT* role to allow at most one occurrence of an actor to its right.

With the word *sell* the following sentences are possible (again within a limited alphabet):

- I sell that.
- I sell furniture.
- I sell her furniture.
- I sell her that furniture.

Here *sell* is the head of the sentence. To its left it has a subject in nominative form which is animate. To its right there is a direct object in accusative

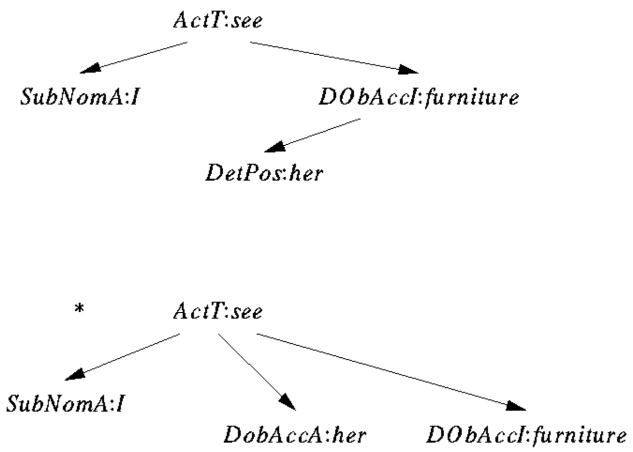

Figure 3

form which is inanimate. In some cases there is also to its right an indirect object, in dative form, which is animate. The third sentence is ambiguous. A casting system $\mathbf{S_2}$ which yields appropriate trees for these sentences (in particular two trees for the second sentence) is in the following table.

Casting system $\mathbf{S_2}$	
$SubNomA \setminus ActT : sell$	$ActT : sell \ / \ DobAccI$
	$ActT : sell \ / \ IobDatA$
$\iota \setminus SubNomA : I$	$SubNomA : I \ / \ \iota$
$\iota \setminus DobAccI : that$	$DobAccI : that \ / \ \iota$
$\iota \setminus DetDem : that$	$DetDem : that \ / \ \iota$
$\iota \setminus DobAccA : her$	$DobAccA : her \ / \ \iota$
$\iota \setminus IobDatA : her$	$IobDatA : her \ / \ \iota$
$\iota \setminus DetPos : her$	$DetPos : her \ / \ \iota$
$\iota \setminus DobAccI : furniture$	$DobAccI : furniture \ / \ \iota$
$DetPos \setminus DobAccI : furniture$	
$DetDem \setminus DobAccI : furniture$	

Unfortunately with this casting system the sentence *I sell that furniture* has two associated dependency trees, just as *I see her furniture* had before. They are shown in figure 4. Again, one is correct, the other one is unacceptable and should be avoided. Likewise *I sell her that furniture* has two trees, one of which is inappropriate because it gives both *her* and *that* a determiner-role w.r.t. furniture. Note that in this case the simple solution of restricting the number of actors to the left of *furniture* will work, but restricting the number of descendants to the right of *sell* does not. It must be possible to have two actors there, as the sentence *I sell her furniture* shows.

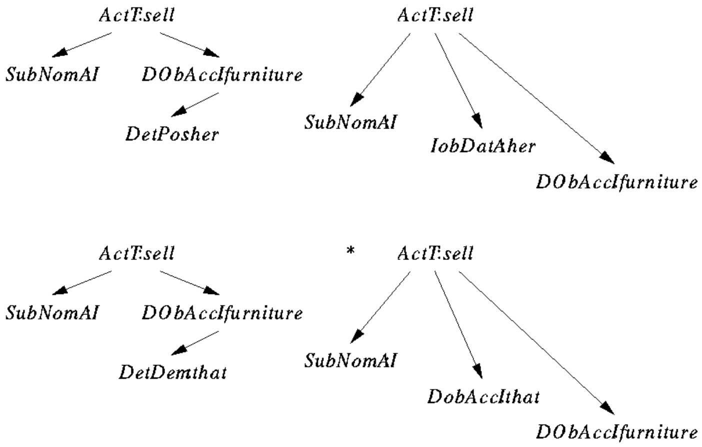

Figure 4

First Modification

As a first step towards a solution of the problem of 'over-ambiguity' illustrated above, we modify casting systems by introducing the possibility of restricting the number of descendants to the left and to the right of a node in the dependency tree. We already indicated this possible modification when we discussed the *see*-examples. The precise modification is as follows. In the definition of a casting system we change the can-be-played-by relation. It was a two place relation between actors and roles, it now becomes a four place relation, in which two intervals are associated with every acceptable role actor combination. So instead of $A : a$ to express that actor a can play role A, we shall now find '$[k,l]A : a[m,n]$'. This still expresses that role A can be played by actor a, but it says more. At the same time it expresses that this role-actor combination will allow a number of descendants to the left that lies between k and l, and a number of descendants to the right lying between m and n. The definition of a dependency tree is changed according to the intended interpretation of the new can-be-played-by relation. That is to say: for nodes $(P, \alpha), (Q_i, \beta_i)$ with (Q_i, β_i) a descendant of (P, α) and occurrence β_i to the left (right) of occurrence α, the range of i is restricted. The i should be in the left- (right-) interval as found in the can-be-played-by relation of the casting system. This restriction is represented in figure 5.

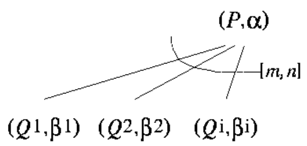

Figure 5

For our two casting systems $\mathbf{S_1}$ and $\mathbf{S_2}$, this leads to the following changes:

Changes of $\mathbf{S_1}$	**Changes of $\mathbf{S_2}$**
[1,1] *ActT : see* [0,1]	[1,1] *ActT : sell [1,2]*
[0,0] *SubNomA : I* [0,0]	[0,0] *SubNomA : I [0,0]*
	[0,0] *DobAccI : that* [0,0]
	[0,0] *DetDem : that* [0,0]
[0,0] *DetPos : her* [0,0]	[0,0] *DetPos : her* [0,0]
[0,0] *DobAccA : her* [0,0]	[0,0] *DobAccA : her* [0,0]
	[0,0] *IobDatA : her* [0,0]
[0,1] *DobAccI : furniture* [0,0]	[0,1] *DobAccI : furniture* [0,0]

The first statement on the left says that *See* must have one argument to the left and can have zero or one arguments to the right. *Sell* must have one argument to the left and can have one or two arguments to the right. This still allows illegal sentences like:

- I sell her (with *her* in a dative role).
- I sell that furniture her (likewise with *her* as in *to her*).
- I sell that furniture furniture (with both occurrences of *furniture* as direct object).

Worse even, it also still allows two dependency trees for *I sell that furniture*, where only one is appropriate.

Second Modification

Another possible modification concerns the left- and right-combine relations of a casting system. Originally they are three place relations between two roles and an actor, we can change them into four place relations, adding an interval of admissible numbers. So, instead of $A : a/B$ to express that actor a in role A can be combined with an actor in role B to its right, we shall now find something like '$A : a/[k,l]B$'. This says not only something about the possibility of combining a, A and B, but at the same time it restricts the 'multiplicity' of that combination: the number B's involved must be in the interval $[k,l]$. Of course the left-combine would be changed analogously. The definition of a dependency tree is adapted according to the intended interpretation of the new combine relations. I.e. for nodes $(P,\alpha),(Q,\beta_i)$ with (Q,β_i) a descendant

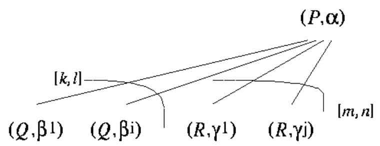

Figure 6

of (P, α) and occurrence β_i to the left (right) of occurrence α, the range of i is restricted. It should be in the interval for the combination of P, α and Q. Figure 6 shows this restriction. For the example this leads to:

Changes to S_1	**Changes to S_2**
SubNomA [1,1] \ *ActT : see*	*SubNomA* [1,1] \ *ActT : sell*
ActT : see / [0,1] *DobAccA*	*ActT : sell* / [0,1] *IobDatA*
ActT : see / [0,1] *DobAccI*	*ActT : sell* / [1,1] *DobAccI*
DetPos [0,1] \ *DobAccI : furniture*	*DetPos* [0,1] \ *DobAccI : furniture*
	DetDem [0,1] \ *DobAccI : furniture*

This specifies that the verb *see* must have an animate nominative subject to the left and can have an accusative direct object, animate as well as inanimate, to the right. The verb *sell* must have a an animate nominative subject to the left and an inanimate accusative direct object to the right. It can have an animate dative form to the right as well. The noun *furniture* can have a determiner to its left, this can be a demonstrative or a possessive form. This system allows illegal sentences like:

- I see furniture her.
- I sell that her.
- I sell that her furniture.

Worse even, we still find correct sentences with two dependency trees, where only one is appropriate:

- I see her furniture (both *her* and *furniture* as direct objects of *see*).
- I sell her that furniture (both *her* and *that* as determiners for *furniture*).

The problem of the two trees for *I sell her furniture* is solved!

Combination of Ideas

The most fruitful approach seems to lie in a combination of the previous two in a uniform framework. The basic idea of the framework is the introduction of the notion of *support set.* Rather than specifying in a casting system something like: actor a in role A can play together with any actor in role B to its right (‘$A : a/B$’), we shall specify something like actor a in role A can play with any actor to its right which has a role from the set $\boldsymbol{B}$: ‘$A : a/\boldsymbol{B}$’. Such a set $\boldsymbol{B}$ in this context is a support set. It is a set of roles, which takes the place of an individual role in the combine relations. E.g. we could introduce a single maybe large set of “see-ables” to be found in the right-combine relation with the actor *see.* Likewise we could introduce two support sets for the right-combine relation of *sell*: the set of “buyers” and the set of “sell-ables”.

Once we have combine relations with support sets, we assign intervals to them as we did in the second alternative discussed above. In fact, that second alternative is a special case, one could see it as a situation where support sets had just a single element. But at the same time the first alternative is a special case of the new approach. Restricting the number of roles that can occur to the right or to the left of a given role-actor combination is like assigning an interval to the full left- and right-support sets. Full here means: containing every role that could possibly occur.

One could say that assigning intervals to combine relations which have support sets rather than individual roles, gives us the best of both worlds. We can restrict the total number of arguments that goes with an actor, by introducing a large support set with all possible arguments and giving it the appropriate interval. At the same time we can exclude cooccurrence of conflicting roles as arguments (e.g. twice the same role) by selecting the proper support sets and restricting the number of occurrences for members of that set.

To give an example, consider the following right-combine clauses.

$$A : a[0,1]/\{X,Y\}, A : a[0,1]/\{Z\}, A : a[1,2]/\{X,Y,Z\}.$$

Together they express that the following five possibilities exist for combining actor a in role A with other roles to its right: there may be a single X, a single Y or a single Z, or there is a combination of two roles (in any order), namely XZ or YZ. What is impossible to the right of $A : a$ is one of the following: no supporting roles at all, support by more than two roles, and support by one of the four combinations of two roles XX, YY, ZZ, and XY.

For the example casting systems we would have the following changes:

Changes to $\mathbf{S_1}$	**Changes to $\mathbf{S_2}$**
$\{$*SubNomA*$\}$[1,1] $\backslash$ *ActT : see*	$\{$*SubNomA*$\}$[1,1] $\backslash$ *ActT : sell*
ActT:see/[0,1]$\{$*DobAccA,DobAccI*$\}$	*ActT : sell* / [0,1]$\{$*IobDatA*$\}$
	ActT : sell / [1,1]$\{$*DobAccI*$\}$
$\{$*DetPos*$\}$[0,1]$\backslash$*DobAccI : furniture*	$\{$*DetPos,DetDem*$\}$[0,1]$\backslash$*DobAccI:furniture*

Now all problems with the original problem cases are eliminated. It is not clear whether or not this method will give a solution for every conceivable problem regarding the number aspect of valency, but it certainly is a step forward. Situations which may still pose problems are addressed in Kersten (1994). In that paper, the ideas are combined 'straightforward' and not through support sets.

4.2 Casting Systems and Dependency Trees with Restrictions on the Number of Arguments

What remains to be done of course is to formalize the foregoing in a new definition for casting systems and dependency trees. First we give a formal definition of an interval.

Definition

An *interval* in the context of casting systems is a set of natural numbers with the property that if it contains k and l, then it contains every natural number between k and l.

In casting systems, we have in fact two kinds of intervals. The closed ones, with a minimal and a maximal element, and the open ones, which have a minimal element but no upper bound. The two will be denoted by $[k, l]$ and $[m, \infty]$ respectively, where $l \geq k$ and $k, m \geq 0$.

The new definition of a casting system now becomes:

Definition
A *casting system* $\mathbf{S}$ is a six tuple with the following components:

- $\boldsymbol{A}$, the *actor* set of $\mathbf{S}$. $\boldsymbol{A}$ is a finite alphabet. Its elements are actors.
- $\boldsymbol{P}$, the set of *roles* of $\mathbf{S}$. $\boldsymbol{P}$ is a finite set.
- $\boldsymbol{L}$, the set of *leading roles* of $\mathbf{S}$. $\boldsymbol{L}$ is a subset of $\boldsymbol{P}$.
- $- : -$, the '*can-be-played-by*' relation of $\mathbf{S}$. It relates roles and actors. If P is a role and a is an actor then we write $P : a$ to express that P can be played by a.
- $- [,] \backslash - : -$, the '*can-be-combined-left*' relation of $\mathbf{S}$. It relates roles and actors with intervals and sets of roles. If P is a role, $\boldsymbol{Q}$ is a set of roles, $[k, l]$ is an interval and a is an actor, then we write $\boldsymbol{Q}[k, l] \backslash P : a$ to express that a in role P can play together with actors in roles from the set $\boldsymbol{Q}$ to its left. The number of such actors is restricted and must lie in the interval $[k, l]$.
- $- : - /[,] -$, the '*can-be-combined-right*' relation of $\mathbf{S}$. The counterpart of the previous relation in the following sense: we write $P : a/[k, l]\boldsymbol{Q}$ to express that a in role P can play together with actors in roles from the set $\boldsymbol{Q}$ to its *right*, with the same restriction on the number of such actors as above.

The new definition of a dependency tree is as follows.

Let $\mathbf{S}$ be a casting system with actor set $\boldsymbol{A}$, and let $\underline{u}$ be a string of actors. A casting tree or a dependency tree for $\underline{u}$ w.r.t. $\mathbf{S}$ is a directed graph $\underline{T}$. The nodes of $\underline{T}$ are pairs (P, α), with P a role of $\mathbf{S}$, and α an occurrence of an actor of $\mathbf{S}$ in $\underline{u}$. The graph $\underline{T}$ has the following properties:

- It is a tree.
- The role of every node can be played by the actor of the node.
- Let (P, α) be an node of $\underline{T}$ and let a be an occurrence of the actor a. For every left-combine relation $\boldsymbol{Lc}[k, l] \backslash P : a$ of $\mathbf{S}$, the number of left-descendants of (P, α) of the form (Q, β), with Q an element of $\boldsymbol{Lc}$, must be in the interval $[k, l]$. The same holds (symmetrically) for the right-descendants of (P, α) and the right-combine relations for this role-actor combination.

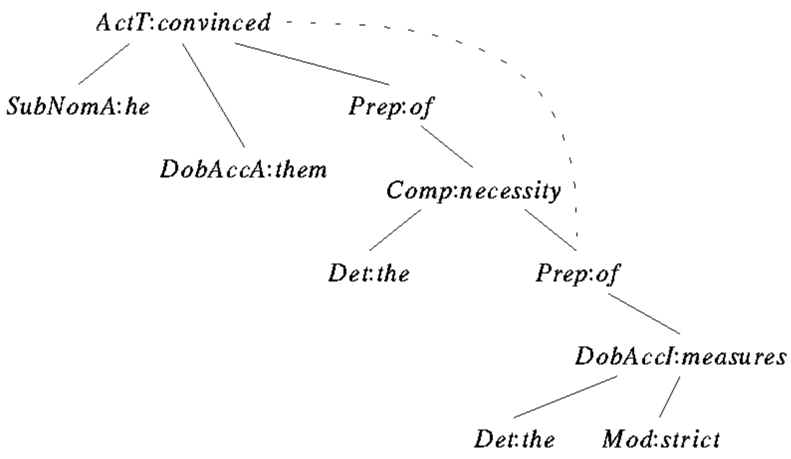

Figure 7

- If there are no support set $\boldsymbol{Ls}$ and interval $[k,l]$ with $\boldsymbol{Ls}\ [k,l]\backslash P : a$ in the left-combine relation of $\mathbf{S}$, than node (P,α) (where α is an occurrence of a) has no successors (Q,β) with β to the left of α, the same holds for the right-combine relations and successors to the right.
- With every node there is a segment $\underline{v}$ of $\underline{u}$, which consists of the actors in the node and in its descendants. In particular, the root-node corresponds to the entire sequence $\underline{u}$.

Note how with this new definitions the invisible role has disappeared. To express that an actor in a specific role simply cannot have a descendant, one just leaves the role-actor combination from the combine relations. To express that there could be descendants, but that they are not necessary, one assigns an interval $[0,l]$ to the combine relations for this role-actor combination.

A final formal observation, slightly outside the scope of this paper, is that the associated family of languages for the new casting systems, is the family of context-free languages.

5 Consequences and Conclusions

We will show two sentences and the correct and incorrect analyses of the sentences by a casting system. Next we will show that the incorrect analyses can be filtered out by means of valency.

- He convinced them of the necessity of the strict measures.
- He spotted the school bus.

The first sentence contains two occurrences of the actor *of*, where the first occurrence should be connected to *convinced*, and the second occurrence to

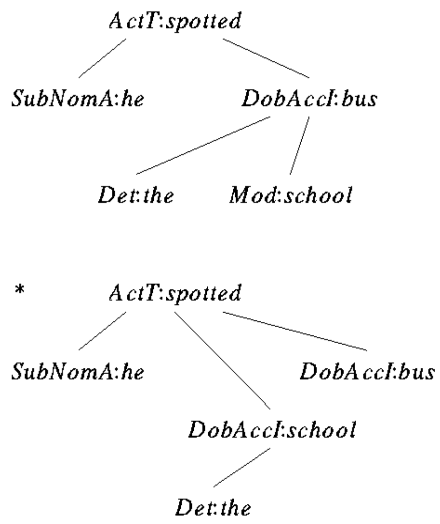

Figure 8

necessity. Figure 7 shows the correct analysis of the first sentence as well as the incorrect one, where the second occurrence of the word *of* is connected to the actor *convinced.* This problem is solved when we specify that to the right of *convinced* in the role *ActT* we can find at most one preposition (role *Prep*).

The second sentence contains two nouns, which both can occur to the right of the actor *spotted.* That is, in a simple sentence like "*he spotted the school*", we can find the actor *school* to the right of the actor *spotted* in the role *ActT.* The analyses of the second sentence are shown in figure 8. Here we see that two objects can be connected to (the right of) the verb, where only one object should be present. Once more the problem can be solved by use of valency.

Implementation of the number aspect of valency in casting systems could help to select the best dependency tree. In Kersten (1994) we find that casting systems enhanced with valency, have the same (mathematical) properties as context-free grammars. In this paper, as mentioned before, the ideas are combined differently. It will not be very difficult though to proof that the system in this paper shares this property.

We mentioned that the addition of valency to casting systems often enables us to choose the best (or the correct) analysis of a sentence. If valency is implemented in the process of *building* casting trees, this should improve the efficiency of the process (which might be interesting if the dictionary is large). The ideas are not yet implemented in a casting system.

References

Appel, R. (1992). *Inleiding algemene taalwetenschap.* Dordrecht: ICG Publications. pp. 101–105.

Hoeven, G. F. v. d. (1992). An experiment in the syntactical analysis of english noun phrases. Memoranda Informatica 92-94, University of Twente, Enschede.

Kersten, D. (1994). Properties of casting systems. Master's thesis, University of Twente, Enschede.

Schubert, K. (1987). Metataxis – contrastive dependency syntax for machine translation. In *Distributed language translation*, vol. 2. Dordrecht: Foris Publications.

Tesnière, L. (1953). *Esquisse d'une syntaxe structurale.* Paris: Klinksieck.