

Meaning and Form in Event Calculus

MSc Thesis (*Afstudeerscriptie*)

written by

Edgar J. Andrade-Lotero

(born December 21st, 1978 in Bogotá, Colombia)

under the supervision of **Prof Dr Michiel van Lambalgen**, and submitted
to the Board of Examiners in partial fulfillment of the requirements for the
degree of

MSc in Logic

at the *Universiteit van Amsterdam*.

Date of the public defense: **Members of the Thesis Committee:**
January 30th, 2006

Prof Dr Arie Verhagen
Prof Dr Martin Stokhof
Prof Dr Peter van Emde Boas
Prof Dr Michiel van Lambalgen
Dr Theo M.V. Janssen



INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Contents

1	Introduction	6
1.1	The Proper Treatment of Events	6
1.2	Cognitive Semantics	7
1.3	EC and Temporal Discourse	8
1.4	This Thesis	9
1.4.1	Part I	9
1.4.2	Part II	9
1.4.3	Part III	10
I	From DRT to EC	11
2	Discourse Representation Theory (DRT)	12
2.1	The DRT-Translation Algorithm	13
2.1.1	Easy first example	13
2.1.2	The general form	15
2.1.3	The rule for Proper Names	16
2.2	Tensed Sentences in DRT	16
2.2.1	Eventualities	17
2.2.2	Tense	17
2.2.3	Aspect	17
2.2.4	Temporal perspective	18
2.2.5	Features for tense and aspect	19
2.3	Past Tense Sentences in DRT	20
2.3.1	Simple past tense	21
2.3.2	Consecutive past sentences	23
3	Event Calculus (EC)	28
3.1	The Language of EC	28
3.2	Eventualities from Predicates	29
3.2.1	Gödel numbering	29
3.2.2	Feferman coding	29
3.2.3	Events and fluents with parameters	30
3.3	Constraint Logic Programming	30

3.4	Integrity Constraints	32
3.5	Aktionsart	34
3.6	IC's for Tensed Sentences	36
3.7	Semantic Representation in EC	37
4	Translating from DRS	42
4.1	Translation of an Example	42
4.1.1	Identification of the language	42
4.1.2	Construction of the scenario	44
4.1.3	Construction of the IC's	45
4.2	Test on Other Past Tense Sentences	51
4.3	Comments on DRT and EC	54
II	Construction Grammar and Event Calculus	56
5	Cognitive-Functional Linguistics	57
5.1	The New Psychology of Language	57
5.2	The New 'Atoms': Constructions	58
5.3	Syntax and Semantics	60
6	Construction Grammar (CG)	63
6.1	Argument Structure Constructions	63
6.1.1	Construction Grammar	64
6.1.2	A comparison with lexicosemantic rules	64
6.2	Verbs and Constructions	67
6.2.1	Verb meaning	67
6.2.2	Construction meaning	68
6.2.3	Participant roles	69
6.2.4	Argument roles	69
6.3	Role Linking Principles	69
6.4	Possible Relations Between Verbs and Constructions	72
7	CG and EC	75
7.1	Ditransitive Construction in EC	75
7.2	Using EC Ditransitive Construction	77
7.3	The Caused Motion Construction in EC	79
7.4	Using EC Caused Motion Construction	84
7.5	An Improvement Using Grammar	85
7.5.1	Kay's attempt at formalizing CG	85
7.5.2	Integration of Kay's attempt with EC	88
7.6	Comments on CG and EC	89

III	A Brief Discussion	92
8	Cognitive Models and Truth Conditions	93
8.1	Cognitive Models and Semantics	93
8.2	EC Truth Conditions	95
8.2.1	Formal tools	95
8.2.2	Truth	97
9	Appendix — Some details of EC	99
9.1	The Language of EC	99
9.2	The Axioms of EC	101

Acknowledgements

I would like to thank my supervisor Michiel van Lambalgen for his very valuable and insightful guidance throughout the writing of this thesis. To Martin Stokhof and Fritz Hamm, for valuable answers, comments, advices, and encouragement. To my family and my girlfriend, for their moral and emotional support. To all my professors and fellow students at the ILLC, for the amazing and enriching academic environment they have created. Finally, to Nuffic and the ILLC, for partial financial support throughout my MSc studies at the University of Amsterdam.

Abstract

This thesis is two fold. The first part deals with an existence proof of a formal algorithm that takes a narrative discourse and returns its semantic representation in the Event Calculus (EC from now on) [19]. This algorithm uses Discourse Representation Theory, and takes on the form of a translation from Discourse Representation Structures to Scenarios and Integrity Constraints in EC. The second part, under a Cognitive-Functional approach, deals with the closely intertwined tasks of providing tools for the formalization of Construction Grammar, and to give the first few steps towards a more flexible algorithm for the semantic representation in the EC.

Keywords: Event Calculus, Discourse Representation Theory, Construction Grammar, Formal Semantics, Cognitive-Functional Linguistics.

Resumen

La presente tesis tiene dos objetivos. En la primera parte se trabaja una prueba de existencia de un algoritmo formal que toma una narración y regresa su respectiva representación semántica en el Cálculo de Eventos (CE por brevedad) [19]. Este algoritmo usa la Teoría de Representación de Discursos, y consiste en la traducción de Estructuras de Representación de Discursos en Escenarios y Restricciones de Integridad en CE. La segunda parte, bajo un enfoque Cognitivo-Funcional, trata las tareas cercanamente interconectadas de proveer herramientas para la formalización de la Gramática de Construcciones, y de dar los primeros pasos hacia un algoritmo más flexible para la representación semántica en el CE.

Palabras Clave: Cálculo de Eventos, Teoría de Representación de Discursos, Gramática de Construcciones, Semántica Formal, Lingüística Cognitivo-Funcional.

Chapter 1

Introduction

1.1 The Proper Treatment of Events

Michiel van Lambalgen & Fritz Hamm [19] developed a formal system with which to analyze temporal discourse. Here, temporal discourse is understood as the linguistic coding of temporal notions. They start out with the assumption that we can gain much insight into temporal discourse by looking at the way humans construct time. They argue that planning and causality are key notions that shape the way humans construct time, and, therefore, temporal discourse can be studied with the help of the notions of planning and causality. The formal system they developed, which will be called here Event Calculus (or EC for short), is intended to be a cognitive representation of causality and planning, and is used to provide an analysis of temporal discourse.

The framework under which this enterprise is carried out is that of Cognitive Semantics (see §1.2). They add a further assumption to this approach: The semantic representation arrived at in the explanation should be computational. Accordingly, they assume that the meaning of a sentence is a cognitive model, where a cognitive model is nothing else but an algorithm. In this particular case, an algorithm is a computational procedure that updates previous declarative information. The previous declarative information is intended to be the background or context against which a sentence is interpreted, and the update of this information is intended to be the cognitive representation of that sentence.

The main achievements of their formal system is the working out of the order of events—which does not come straightforwardly from the order of the sentences—and the coherent representation of different kinds of temporal eventualities—fluents and events.¹

In the following section I will sketch the framework of Cognitive Semantics, in order to get to grips with van Lambalgen's and Hamm's cognitive approach.

¹See, for instance, [19, Ch. 9].

1.2 Cognitive Semantics

Given any natural language L , it is the job of the syntactician to come up with an appropriate set of rules that describe the set of concatenations of lexical elements that are licensed by (the linguistic community of speakers of) L , i.e., the set of sentences of the language.

It is the job of the semanticist to come up with a specification of the meanings of the expressions of a (subset of a) language. Modern Logical Semantics is based on the Correspondence Theory of Meaning,² also dubbed the referential approach to meaning, in which meanings of expressions are regarded as associations between the expressions and something else. The nature of that ‘something else’ divides this approach to meaning in two: on the one hand, we have Formal Semantics, in which ‘something else’ are things in the outer world³; on the other hand, we have Cognitive Semantics, in which ‘something else’ are concepts in our minds. It is argued in this type of approach that there are no things in the outer world that can be thought of as referents for the expressions. This is not to deny the existence of the outer world, it is just to say that the referents of our expressions are conceptualizations of the outer world. The main points of this approach are brought out by Jackendoff [8], to which I turn to discuss now.⁴

E-Language and I-Language The first concepts that need to be explained are the concept of E-Language and I-Language. The former is thought of as a tool, an artifact that humans use and whose properties make part of the ‘external’ world. The latter is a psychological concept, a body of knowledge that resides inside the mind of the speakers.

E-Semantics and I-Semantics The distinction drawn between E-Language and I-Language has also a counterpart at the semantic level. Accordingly, E-Semantics is the description of the relation between E-Language and the ‘external’ world. I-Semantics is the description of the relation between I-Language and *our conceptualization* of the ‘external’ world.

Grammaticality and Truth The concepts of a sentence being grammatical/true are relativized in accordance with the I-Language and I-Semantics concepts:

E-gram String \mathcal{S} is a grammatical sentence of language L .

I-gram A speaker of language L judges string \mathcal{S} grammatical.

²Cf. [4, Ch. 1, Vol. 2].

³Cf. [4], [12].

⁴It might be worth saying that I am not committed to this approach at a personal level. I believe that it has a number of problems with respect to the nature of the explanation of meaning. Nonetheless, I will completely work here under the assumptions of Cognitive Semantics. A brief discussion of my qualms are presented in chapter 8.

E-true Sentence \mathcal{S} of language L is true iff⁵ conditions C_1, \dots, C_n obtain in the world.

I-true A speaker of language L judges sentence \mathcal{S} true iff conditions C_1, \dots, C_n obtain in his construal of the world.

The Nature of Reference and Meaning The concept of reference is internalized: words refer to our conceptualization of the ‘external’ objects. Let us consider Jackendoff’s example:

(1.1) The meeting was changed from Tuesday to Monday.

The concept of reference of the terms in (1.1) cannot be taken realistically: ‘this sentence points to nothing perceptible. It describes something that takes place only in people’s minds, changing their future behavior ... [By the same token, T]here is nothing *in the world* that demands a sharp boundary between red and orange, and there is nothing *in the world* that sharply distinguishes climbing from other kinds of locomotion.’⁶

1.3 EC and Temporal Discourse

The formalism of Event Calculus —developed initially by Murray Shanahan [15]— represents actions and their effects. This is a formalism widely used in AI, but only recently has it been used in linguistics —mainly, by M. van Lambalgen and F. Hamm.⁷ Several formal tools were added to EC of Shanahan in the pursue of coping with linguistic data on tense and aspect. Among others, the formal tools added are the Feferman coding —for the introduction of parameters into fluents and events—, Constraint Logic Programming —in order to constraint the class of models—, and Integrity Constraints —in order to code the tense of the sentences via goals and planning. This formalism —widely explained in [19]— is briefly explained in §3.

The scope of analysis of the formalism is temporal discourse. We understand here a temporal discourse as a sequence of sentences describing a series of eventualities. It turns out that the concept ‘discourse’ is necessary —i.e., the unit of analysis is not the sentence but sequences of sentences—, because several phenomena, for instance ‘temporal anaphora’, ‘temporal perspective point’, and tenses like ‘Imparfait’ in French, only make sense in a temporal discourse. The analysis of tense and aspect would be mutilated if analyzed at the sentence level. Thus, the goal of EC is the explanation of the order and structure of the eventualities described in a sequence of sentences.

The explanation of the semantics of pieces of temporal discourse provided by EC is as of yet informal. If we were to analyze an example like:

⁵Throughout this thesis, and in accordance with common mathematical usage, I will use ‘iff’ as a short for ‘if and only if’.

⁶[8, p. 557]. Italics are his.

⁷Cf. [18] and [19].

(1.2) Jean took off his sweater. The room was warm.

we would come up with a scenario and integrity constraints for each sentence, following an update of the scenario brought about by these integrity constraints.⁸ However, the theory by itself does not provide us with any reasons as to why these scenarios and integrity constraints, and no others, have to be used in this case. Further, it does not provide us with a systematic method for producing the scenario and integrity constraints in question: we come up with these in an informal way. It is the task of the first part of this work to argue that such reasons can be given, and that the link between natural language temporal discourse and semantic representation can be carried out in a formal way.

I should make clear the scope of the analysis given in here. I restricted my attention only to past tense sentences. However, I believe that a similar analysis can be given to other tenses as well, like the past and present progressive, and the present tense. (Though, the future tense might represent a problem.)

1.4 This Thesis

1.4.1 Part I

In the first place, I give a formal (theoretical) algorithm that goes from the syntactic representation of simple past tense sentences in English to a semantic representation in EC. This algorithm is rather complex, since the idea is to translate Discourse Representation Structures —the semantic representations provided by Discourse Representation Theory— into semantic representations in EC.⁹ In order to explain the whole algorithm, then, we need to get to grips with Discourse Representation Theory. This is the purpose of Chapter 2. In chapter §3, I explain EC formalism, in which I introduced some changes. The reason for this changes is to make it easier to handle with the formal definition of aktionsarten (Cf. §3.5). Chapter 4 is the core of part I, since it deals with the translation between DRT and EC. This section ends with some comments about this algorithm, further ways to improve it, and some reasons to look for different algorithms.

1.4.2 Part II

In §4.3 some arguments for improving the translation attempted in the first part are given. But instead of focusing on a direct improvement of the algorithm, we will change our approach to the problem. Some comments are in place here. In the first part, we attempted a translation that is based on DRT. This latter starts from syntax, so the semantic representation of a temporal discourse is rather indirect. We first process the discourse syntactically, then we give a

⁸Cf. §3.7 for the treatment of this example in the actual state of affairs, and Cf. §4.1 for the treatment of this example starting out from syntax —i.e., in the improved way proposed in this work.

⁹This idea was first given in [7], to which this work owes a good amount of insightful ideas.

semantic representation in DRT and then we transform this into a representation in EC. This presupposes that each step is independent from the next one.¹⁰ In the second part of the thesis, I will explore a semantic representation based on a ‘cognitive-functional’ approach (Cf. §5). This approach overrides the order of steps just mentioned. In the long term, we expect this change to bring better results than a direct improvement on the translation given in the first part.

The first chapter of the second part, §5, is an introduction to the new approach to cognitive linguistics. Since it differs from the framework used to give the translation algorithm in the first part, I will make a short discussion of some points that change along with the approach. The principal change brought about by the change in approach is the change in the underlying syntactic theory. That is, we will move onto Construction Grammar. An introduction to this theory is given in chapter §6, in which I present a brief summary of the classical defense of argument structure constructions, based on [5]. The next chapter, §7, presents the attempt to formalize two argument structure constructions meaning by means of EC. A discussion of what was achieved and what remains to be done is given in §7.6.

1.4.3 Part III

The last part of the thesis (i.e., Chapter 8) is a brief discussion about cognitive models and truth conditions. The discussion intends to be a presentation of my personal qualms regarding cognitive models. The remedy to these qualms is a move towards truth conditions, move that is briefly explored in this chapter.

¹⁰This independence is true up to certain extent. Each step requires something particular from the previous one, and to this extent there exists certain dependence.

Part I
From DRT to EC

Chapter 2

Discourse Representation Theory (DRT)

The main objective of this part of the thesis is to present *a* formal way to translate natural language temporal discourse into a semantic representation in EC. Suffice it to define a temporal discourse as a sequence of sentences describing eventualities. It would be enough for our purposes to find a theory that delivers appropriate semantic representations to temporal discourse, and which can be translated into semantic representations in EC. Discourse Representation Theory¹ (DRT from now on) is a semantic theory that allows us to do so.

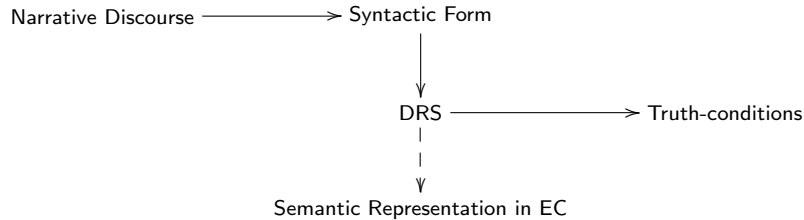
Now, DRT's starting point is syntax (in particular, the syntactic theory called Generalized Phrase Structure Grammar), and syntax provides a syntactic form to each sentence in the temporal discourse.² Thus, the upshot of the syntactic theory is a sequence of syntactic forms. DRT provides an algorithm to transform this sequence of forms into a Discourse Representation Structure (DRS from now on, see below) in an incremental way. That is, it builds up a DRS for the first sentence and adds the new information represented in the second sentence to this DRS, and so on. A DRS is regarded as a discourse representation, and it fits into the Realistic Formal Semantics paradigm (RFS from now on) because we can give a precise definition of the truth conditions for a DRS, and thus to the syntactic representation of each sentence of the temporal discourse. It is not relevant for us to go into the discussion of whether DRT fits completely or partially into the picture given by either RFS or Cognitive Semantics.³ Rather, my interest in DRT comes from the possibility to give a formal translation from DRS into EC. This implies that, if we could come up with a formal way to transform a DRS into a semantic representation in EC, we would have a formal way of going from (the syntactic representation of)

¹My main sources to DRT are [9] and [4, Vol 2.].

²As long as these sentences are in the domain of the syntactic theory. The details are not important here.

³But see [4], [16], and [7] for a discussion.

temporal discourse to a semantic representation in EC.⁴ The important ideas are summed up in the following figure:



This chapter deals, in a very general way, with the solid downward arrow, i.e., the path from Syntax to DRT. Chapter 4 deals with the dashed downward arrow, i.e., the path from DRT to EC.

2.1 The DRT-Translation Algorithm

As I said before, the starting point to give a temporal discourse a semantic representation in DRT is syntax.⁵ In DRT, there is an important modification of the syntactic information. That is, the construction rules (see below) modify the syntactic tree in such a way that it does not fit anymore into the original syntactic theory. This will become clearer as long as we go into the details of the algorithm.

To begin with, we need to define a Discourse Representation Structure (DRS). A DRS is a box-like semantic representation, which is intended to capture the discourse meaning of a set of sentences S_1, \dots, S_n . It consists of a universe, symbolized by \mathbf{U}_K , and a condition set, symbolized by \mathbf{Con}_K . The elements in \mathbf{U}_K are called discourse referents, and are to be interpreted as objects in a First-Order Model Universe. The elements of \mathbf{Con}_K are to be distinguished in two classes: reducible and irreducible conditions (see §2.1.2). A DRS is complete if it contains only irreducible conditions, which are to be interpreted as predicates in a First-Order Model. I will not explain here the interpretation and truth conditions of a DRS. The interested reader is referred to [9, §1.2] or [4, Vol. 2, §7.4] for details.

2.1.1 Easy first example

Consider the following example:

(2.1) John owns Ulysses.

In order to start the translation of this example into a DRS, we need its syntactic structure, which I present without explanation in figure 2.1.1.

⁴This idea is original from [7]. This part of the thesis proposes a detailed construction of this translation algorithm.

⁵I will hardly go into any detail of syntax here. The reader is referred to [9] for details.

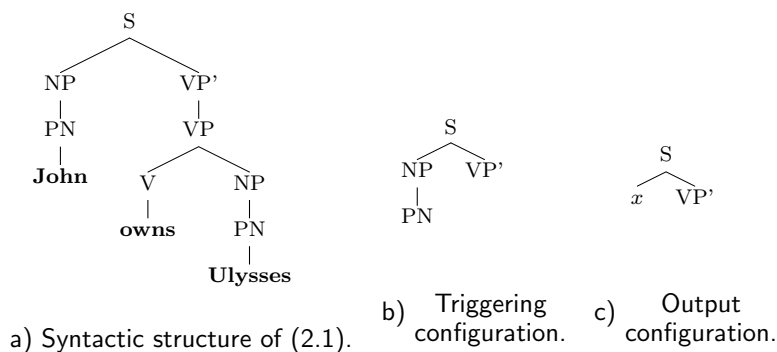


Figure 2.1.1.

We need also an empty DRS, but in general we should assume that this DRS contains all the presuppositions the hearer has and with which he understands the discourse. We should feed this DRS with the syntactic structure of the sentence we want to analyze. We introduce the syntactic structure into \mathbf{Con}_K .

That syntactic structure of our example, now viewed as a condition in the DRS, has a characteristic top-part structure, which determines that the subject is a proper name, namely **John**. This structure is called a *triggering configuration* (see figure 2.1.1). It is a triggering configuration because it is the starting point of a *construction rule*, which modifies the original condition. The construction rule activated by this triggering configuration tells us that we need to introduce a new discourse referent into the universe of the DRS, \mathbf{U}_K . Call this referent x . We should require this discourse referent to refer to **John**, and thus we have to introduce the condition $John(x)$ into the condition set of the DRS, \mathbf{Con}_K . Here it comes an important step. The condition (the original tree in this case) should be modified in order for the algorithm to continue with the rest of the information. So, the outcome of the construction rule is a modification of the input condition, obtained by means of a modification of the triggering configuration (see figure 2.1.1).

The DRS obtained after the application of this construction rule contains two conditions: $John(x)$ and the modified syntactic tree. Two comments are in place here. To begin with, the former condition is called an *irreducible condition*, since no construction rule has it as a triggering configuration — $John(x)$ can be interpreted as a predicate in a First-Order Model without further modification. On the other hand, the latter condition, represented by the modified syntactic tree, is a *reducible condition*, since, as we shall soon see, it contains triggering configurations of some construction rules. It needs some more touches in order for its information to be correctly interpreted in a First-Order Model. The second comment is the following. We said above that the construction rules modify the syntactic tree in such a way that it does not fit anymore in the definition of the syntactic theory. This can now be easily seen, for the tree that appears as a second condition of the previous DRS contains a discourse referent instead of a syntactic category. Each further construction rule will modify even

more this structure, plugging semantic information in it, so to speak. The upshot of the whole procedure: only semantic information remains in the DRS in the form of irreducible conditions. This example is far too simple to be worth continuing any further. I will now turn to the general form of the translation algorithm.

2.1.2 The general form

Here is a quote which clearly states the idea of the construction algorithm:

‘The DRS-construction algorithm involves two *recursions*. The first recursion operates at the level of the complete sentence that the discourse consists of: when the algorithm is applied to a sequence of sentences S_1, \dots, S_n it deals with these sentences in order of appearance. It first incorporates S_1 into the starting DRS K_0 , then it incorporates S_2 into the DRS K_1 resulting from the first incorporation, etc. The first step of the process by which S_i gets incorporated into K_{i-1} consists in adding the syntactic analysis $[S_i]$ of S_i to the set of conditions of K_{i-1} (...) [which] acts as a *context of interpretation* for S_i .⁶

The non-trivial step of the algorithm is to incorporate the syntactic analysis $[S_i]$ of sentence S_i into the set of conditions of K_{i-1} . The analysis consists, in few words, in the reduction of all the reducible conditions in the set of conditions of K_{i-1} plus $[S_i]$. Since we assume that the set of conditions of K_{i-1} does not contain any reducible condition, we only need to reduce the condition $[S_i]$. (From the example in the previous section, we know what reducible and irreducible conditions are.) To fully define the analysis, we need to give the order of application of construction rules, and then we need to give the construction rules.

The order of application of construction rules stands in need of classification. We have to cope with the possibility of there being several reducible conditions to be reduced. This case is rather trivial, since the order is irrelevant. We can reduce the conditions in any order and the outcome will be the same. On the other hand, we also have to cope with the possibility of there being a reducible condition to which we can apply several rules. In this case we have to choose the rule whose triggering configuration (the one that could be applied to the condition) has a node which dominates every other node of the triggering configurations of other candidate rules. If there is no such a rule, we can apply any rule whose top-node is not dominated by any node of other candidate rules.⁷

Now, we need to give a description of the construction rules. The construction rules depend on the syntactic theory that has been chosen. Since the syntax needed to cope with the sentences we are interested in here is rather complex, we would need a large amount of DRS construction rules to deal with all the

⁶[9, p. 85].

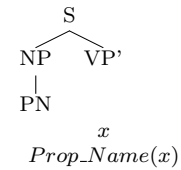
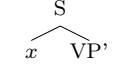
⁷This can be stated in an easier way by means of mathematical terminology. The rules we can apply are those whose triggering configuration have a maximal node in the set of nodes of all the triggering configurations of the candidate rules. As we know from the theory of partial orders, there could be several maximal elements in a partial order. This translates in our example as there being several rules we can apply. The order of application of the ‘maximal’ rules is irrelevant. A further specification: the partial order is defined by the dominance relation in the syntactic tree.

possible cases of syntactic trees. This is clearly out of the scope of this thesis. However, I will give in full detail the explanation of a simple rule: the rule for proper names. The other rules will be introduced as long as we require them. For a more detailed and comprehensive explanation see [9].

2.1.3 The rule for Proper Names

A construction rule starts with a triggering configuration. A triggering configuration is a part of a reducible condition in the set of conditions of a DRS. In the case of proper names, the triggering configuration is presented in figure 2.1.1 and in the table below. Whenever a condition has this structure as one of its parts, we can apply the rule for proper names. This rule stipulates the introduction of a discourse referent, call it x , which will be the referent of the proper name. It also stipulates the introduction of a condition, call it $Prop_Name(x)$. This condition asserts that the discourse referent should refer to the appropriate PN. This requires to search the node below the syntactic category PN within the condition from which we took the triggering configuration. Finally, the rule should modify the reducible condition from which it was triggered. In general, the modification is performed in the triggering configuration. For this case, see figure 2.1.1.

Hereafter, since the description of a rule will follow the previous steps, and these can be easily read off from a standard format (see below), I will omit the explanation of the steps and go straight to the standard format. The standard format for the rule of proper names is:

Construction Rule for Proper Names (CR.PN)	
<p>Triggering Configuration $\gamma \in \text{Con}_K$:</p>	
<p>Introduce into U_K: Introduce into Con_K: Replace γ with:</p>	

2.2 Tensed Sentences in DRT

The aim of this section is to introduce some elements of tense and aspect that are used to deal with the representation of tensed sentences.⁸ The elements introduced in this section cover a range which is broader than we really need, since they embody a more or less coherent whole. It is also necessary to say some few words about the ontology of time which DRT presupposes. Ontology

⁸Though, this presentation is strongly biased towards DRT's interpretation. It is important to bear in mind that there are several ways to present what follows, in particular, EC's presentation differs in several aspects from it. For instance, there is a significant difference in the way that both theories deal with aspectual properties.

of time is paramount to any discussion about tense, and involves the nature of time and events. However, I am not interested in the discussion of the ontology of time per se. The aim here is only to present the particular ontology chose by DRT.

2.2.1 Eventualities

DRT has chosen an “event semantics,” where some temporal entities, called ‘events’, are taken as primitives. They are opposed to ‘instants’ in the sense that the former can be thought of as composed out of the latter. Events like ‘Mary writes a letter’ are not instantaneous, but they have some “duration in time.” Instants refer to non-extended-in-time entities whereas events refer to entities with extension in time. There are three important relations between events: precedence ($<$), overlapping (\circ), and inclusion (\subseteq). Since events are taken as primitives, they are generated by an axiomatization of relations among events. I am not going to reproduce these axioms here (but see [9, p. 667]). I assume that the reader understands these axioms and the consequences that can be drawn from them. On the other hand, there is an important distinction between events and states, but I won’t handle it until section §2.2.3. The term *eventuality* is generally used to refer to both events and states and I will stick to that use.

2.2.2 Tense

Tense is a property of sentences. The main criterion to distinguish between different tenses is the relation between the time of the event referred to by the sentence and the utterance time (\mathbf{n} from now on). Let us denote the eventuality described by the tensed sentence as \mathbf{t} . Accordingly, we can describe the past tense by saying that the eventuality time is previous to the utterance time ($\mathbf{t} < \mathbf{n}$); the present tense by saying that the eventuality time contains the utterance time ($\mathbf{n} \subseteq \mathbf{t}$); and the future tense by saying that the utterance time is previous to the eventuality time ($\mathbf{n} < \mathbf{t}$).

2.2.3 Aspect

Aspectual properties —or *aktionsarten*—, unlike tense properties, are properties of verbs and verb phrases. The aspectual properties taken into account by DRT are *accomplishment verbs*, *achievement verbs*, *stative verbs*, and *activity verbs*. The classification of verbs (or verb phrases) according to aspectual properties depends on two intertwined things. The first thing is whether the eventuality described by the verb can be seen as structuring the time line in three different phases: (I) A preparatory phase that leads to (II) a culmination point, which is followed by (III) a result state. The second thing is —if the eventuality can be so divided— which of these parts are referred to by the past, the progressive, and the perfective sentences related to that verb. To make this clearer, consider the following example with the verb phrase ‘write a letter’.

The eventuality described by this verb phrase does structure the time line in these three phases. The past form, say ‘Mary wrote a letter’, refers to phases I and II. The past progressive, i.e. ‘Mary was writing a letter’, refers to phase I, and the perfective, i.e. ‘Mary has written a letter’, refers to phase III. This kind of verbs are called *accomplishments*. Now, consider verb phrases like ‘reach the top’. This eventuality also divides the time line in three phases. However, the past tense, unlike accomplishments, refers just to phase II. Consider the sentence ‘John reached the top’. If we think of phase I of this eventuality, we cannot say that at that moment John reached the top, because that is something that happens instantaneously. (This previous reasoning might be clearer when carried out with the verb ‘die’, instead of ‘reach the top’.) These verbs are called *achievements*. The characterization of *states* is easy, since they just simply do not divide the time line in these phases. The eventuality described by a sentence like ‘John knew the answer’ cannot be thought of as dividing the time line into a preparatory phase leading to a culmination point. There is no culmination point in this eventuality. John’s state of knowing the answer can completely lay in the past, but it might as well be John’s actual state—he can still know the answer. On the other hand, the classification of *activities* is a little bit odd in DRT. Activities do divide the time line in three phases, but not by themselves. Something more should be added to the VP that describes the eventuality in order to allow for the correct use of some tenses. A sentence like ‘Mary walked’ is odd unless there is some complement that introduces the culmination point, like ‘for one hour’.

The information contributed by aspectual properties in DRT comes down to the use of two syntactic features. These are *STAT* and *PERF*. The former distinguishes between states and events. Therefore *STAT* has two values: *+STAT* (state), and *-STAT* (event). The second feature distinguishes between those verbs which refer to the result state and those which not. Therefore *PERF* has two values: *+PERF* (refers to result state), and *-PERF* (does not refer to a result state).

A notable difference between DRT and EC is that, in the former, the aktionsart does not play an important role, whereas in the latter it is vital. Aktionsarten do not play an important role in DRT simply because the foregoing features—that is, *STAT* and *PERF*—cannot distinguish between aktionsarten. What is important in DRT is which of the three phases—if at all—a sentence refers to. This difference in stress with respect to aktionsarten is one thing that stands in the way of a neat translation between DRT and EC, as we will later on see.

2.2.4 Temporal perspective

In section §2.2.2 we used two eventualities to distinguish among tenses, namely, **t** and **n**. However, when we deal with a sequence of sentences, we need two other eventualities. We are going to call them *reference time*, and *temporal perspective point*. They have to do with temporal anaphora involved in a sequence of sentences. (This is going to be explained in more detail in §2.3.2.) If we consider

one single sentence in isolation, we do not need to think of a reference time nor a temporal perspective point —we need just the time of the eventuality referred to by that sentence—. They can be better appreciated when we interpret a sentence after having interpreted another sentence. The second sentence is interpreted against the background, so to speak, introduced by the first sentence. And we usually interpret the events described in the second sentence as occurring in, or having certain relation with, the ‘vicinity’ of the temporal time introduced by the first one. This ‘vicinity’ is the reference time. In a longer sequence, say of three or four sentences, the reference time can be thought of as either changing from sentence to sentence or to be fixed by the first sentence. The former takes on a form of temporal sequence, whereas the latter takes on a form of an elaboration of, say, a description. On the other hand, the introduction of the temporal perspective in the analysis of tensed sentences is necessary in order to explain the meaning of sentences like:

(2.2) Fred arrived at 10. He had set off at 6.

The eventuality referred to by the second sentence in (2.2) is previous to the reference time introduced by the first sentence. That is, the reference time in which (2.2) is interpreted is given by Fred’s arriving at 10 o’clock. At that time, Fred’s setting off has already happened.

In order to disentangle this, we need three eventualities: utterance time (**n**), reference time (**t**), and temporal perspective point (**TPpt**) —this point is meant to represent the temporal point at which we locate ourselves so as to describe the eventuality. Thus, the first sentence describes a situation at a time previous to the utterance time. It is the time of that situation at which we place the temporal perspective point since it is that time from which we interpret the second sentence. Accordingly, the reference time of the second sentence is previous to the temporal perspective point, since the event of the second sentence is seen in the past at the time described by the first sentence (which already lied in the past).

The relation between reference time and temporal perspective point is controlled by the syntactic feature *TENSE*, which has three values: *past* (**t** < **TPpt**), *pres* (**t** = **TPpt**), and *fut* (**TPpt** < **t**). In order to control the relation between temporal perspective point and utterance time, we need another feature, *TP*, which has two values: *+PAST* (**TPpt** < **n**), and *-PAST* (**TPpt** = **n**).

It is worth noting that when we have a feature like *-PAST*, the feature *TENSE* gives the same information as the one given by §2.2.2.

2.2.5 Features for tense and aspect

By means of the four features presented so far we can describe the following English tenses:

Tense	<i>TP</i>	<i>TENSE</i>	<i>STAT</i>	<i>PERF</i>
present	$-PAST$	<i>pres</i>	$+STAT$	$-PERF$
future	$-PAST$	<i>fut</i>	$+/-STAT$	$-PERF$
simple past	$-PAST$	<i>past</i>	$+/-STAT$	$-PERF$
	$+PAST$	<i>pres</i>	$+STAT$	$-PERF$
past future	$+PAST$	<i>fut</i>	$+/-STAT$	$-PERF$
present perfect	$-PAST$	<i>pres</i>	$+STAT$	$+PERF$
future perfect	$-PAST$	<i>fut</i>	$+STAT$	$+PERF$
past perfect	$+PAST$	<i>past</i>	$+/-STAT$	$-PERF$
past future perfect	$+PAST$	<i>fut</i>	$+STAT$	$+PERF$

2.3 Past Tense Sentences in DRT

[9] argue for an ambiguity in the temporal structure of simple past tense sentences. A first structure is given to sentences like:

(2.3) Mary got to the station at 9:45.

which describes a past event. This event is seen as completely lying in the past, and the perspective from which we see it coincides with the utterance time. The event is in the past and we, at the moment of the utterance, are in the result state after the culmination of the event. This temporal structure is captured with the help of the syntactic features explained in the previous section. To begin with, since we see this eventuality from an actual perspective, the **TPpt** coincides with the utterance time **n**. Plus, the reference time, or time of the eventuality, lies completely in the past, and therefore the reference time is previous to the **TPpt**. The described event with the above temporal properties has the following set up: the feature *STAT* has the value $-STAT$, *TENSE* has the value *past*, and *TP* has the value $-PAST$. Since this eventuality refers to the culminating point, the feature *PERF* has the value $-PERF$.

It is worth noting that the case of a past temporal perspective overlapping the reference time of an event seems almost impossible. However, there are cases where the **TPpt** lies in the past, and it coincides with the reference time. Such cases are motivated by certain behavior of the indexical **now**. Consider the example:

(2.4) Mary had been unhappy in her new environment for more than a year.
But now she felt at home.

The second sentence of (2.4), by means of the indexical **now**, situates the temporal perspective in the past. This sentence does not have the structure of the past perfect, and besides, it is quite intuitive that the temporal perspective coincides with the reference time. Thus, the temporal properties of such a sentence obey the following set up: the feature *STAT* has the value $+STAT$, *TENSE* has the value *pres*, and *TP* has the value $+PAST$. Since this eventuality is a state, *PERF* has the value $-PERF$.

This ambiguity is pretty clear in French. The *Passé Simple* has the characteristics of the first set up, whereas the *Imparfait* has all the characteristics of the second.

There is a third possibility allowed. A state can also be described from an actual temporal perspective and lying completely in the past, as in the example:

(2.5) John knew the answer (but he does not know it anymore).

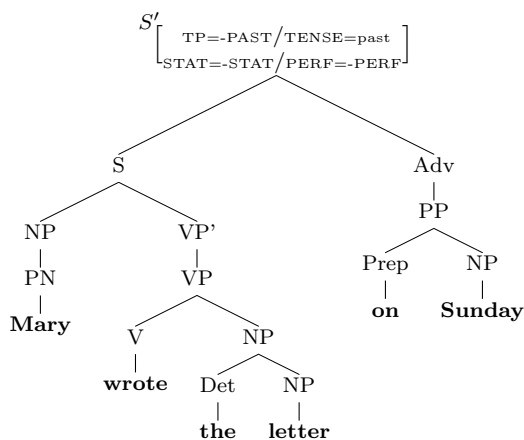
The set up in this case is clear: the feature *STAT* has the value $+STAT$, *TENSE* has the value *past*, and *TP* has the value $-PAST$. Since this eventuality is a state, *PERF* has the value $-PERF$.

2.3.1 Simple past tense

A rigorous way to present the DRT translation algorithm would be to give the syntax for the fragment of English we are interested in, next the construction rules, and finally some examples. The order of presentation here is inverted, mainly for ease of presentation. I am going to start with an example of a simple past tense sentence, and therewith I will introduce the construction rules we will need. The example is:

(2.6) Mary wrote the letter on Sunday.

The algorithm requires a lot on the syntax. One thing it requires is for the top node to bear all the temporal information. A tensed sentence is considered as describing an eventuality. This eventuality should be located with respect to other three eventualities: the utterance time, the temporal perspective point, and the reference time. The latter gives a sort of temporal context, and usually is got from the context given by the previous sentences. Although the relations among these eventualities are naturally thought of as being encoded in the verb, the algorithm requires this information to be accessed in the first step of processing, that is, the top node of the syntactic tree. I am going to assume without explanation the following syntax for example (2.6), and from here I will explain the construction rules needed for it.

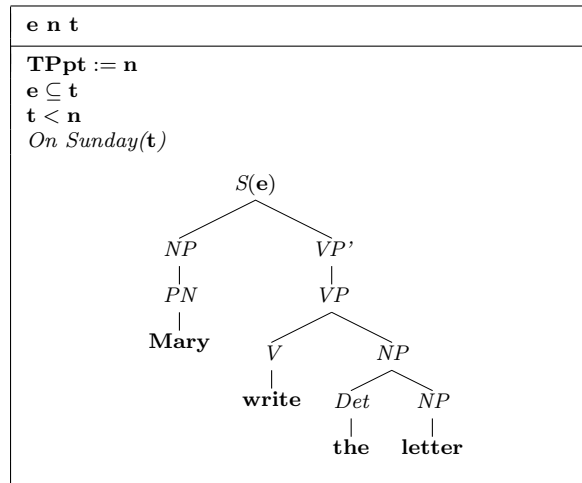


Suppose that we have an empty DRS K and that we introduce into \mathbf{Con}_K the previous syntactic tree. This tree is seen as a reducible condition of the DRS.

This condition contains temporal information for the first step of the process, which is triggered by the configuration embodied by the nodes S' , S , and Adv . From this configuration we are going to represent the following information:

1. The sentence describes an eventuality. Therefore, introduce a new discourse referent \mathbf{e} . Also, introduce the discourse referent \mathbf{n} for the utterance time, and a new discourse referent \mathbf{t} for the reference time.
2. Find the value of the temporal perspective point \mathbf{TPpt} according to TP . Since TP is $-PAST$ in this example, then $\mathbf{TPpt} := \mathbf{n}$.
3. Record the temporal relation between the described eventuality and the reference time according to $STAT$. In this example, the value of $STAT$ is $-STAT$, i.e., the sentence describes an event. The relation is $\mathbf{e} \subseteq \mathbf{t}$.
4. Record the temporal relation between the reference point and the \mathbf{TPpt} according to $TENSE$. The example has got a value *past* for $TENSE$, therefore the relation is $\mathbf{t} < \mathbf{n}$ (recall that $\mathbf{TPpt} := \mathbf{n}$ in this example).
5. Introduce a new condition for the temporal adverb. This condition should be of the form $\beta(\mathbf{t})$, i.e., it constraints the reference time. In the example, the temporal adverb is **on Sunday**. Therefore, we need to introduce in \mathbf{Con}_K the condition $On\ Sunday(\mathbf{t})$.
6. We should modify the condition in such a way to keep track of the discourse referent already introduced with the aim to represent the described eventuality. In this case, such referent is \mathbf{e} . We should, therefore, plug this referent into the condition for further construction rules. Also, and since we already represented the temporal information of the sentence, we should ‘detense’ the remaining parts of the condition; we should modify the condition into an infinitive form.

The application of the first step in our example gives as a result the following DRS:



The next step involves the handling of the subject PN. In order for the rule that we introduced in §2.1.3 to apply in this case we should make a slight modification to it. We should use (**e**) in both the triggering configuration and the modification of the condition. It is worth noting that the information of the discourse referent which represents the described eventuality is carried along from the S node to the VP' node.

The transition between VP' and VP is not important here. Though, it is necessary for the use of auxiliary verbs, like in the future tense. In this case, the only modification brought about by a construction rule that handles with the VP' would be the carrying along of the eventuality (**e**) to the node VP. The next step treats the NP, by introducing a new discourse referent y with the condition *the letter*(y), plus carrying along the eventuality into the V node. The outcome of this three steps in our example is:

e n t x y
e \subseteq t
t $<$ n
<i>On Sunday</i> (t)
<i>Mary</i> (x)
<i>the letter</i> (y)
e : <i>write</i>(x, y)

Note that I modified the condition that carries the remaining part of the tree to the condition **e** : *write*(x, y). This is a much more readable presentation of the DRS and is the one that I am going to adopt here. I will also leave out the condition **TPpt** := α , since this is a redundant information.

2.3.2 Consecutive past sentences

This subsection deals with two much more interesting examples which will allow us to explore the role of the ‘temporal anaphora’ and the ambiguity we have allowed in the simple past tense sentences.

I will analyze two examples of consecutive past tense sentences. The first will be developed in detail, and the second will be built upon the first, giving importance only to the differences with the former. The main difference between them is the following. The former consists of a succession of two event-sentences, whereas the latter is an event-sentence followed by an state-sentence. The first example is:

(2.7) Peter went into his room. He turned on the radio.

I am going to perform the same steps as in the previous example except from the introduction of the temporal adverb. So, this seems to be a good place to introduce the general rule of the top node for initial sentences. Such a rule is presented in table (2.A). In our example, the application of this and the

(CR.S') Construction Rule for the top node $TP = \alpha, TENSE = \beta, STAT = \delta$	
Triggering Configuration $\gamma \in \mathbf{Con}_K$:	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> (i) $\begin{array}{c} S' \\ \\ S \\ \wedge \end{array}$ </div> <div style="text-align: center;"> (ii) $\begin{array}{c} S' \\ / \quad \backslash \\ Adv \quad S \\ \wedge \end{array}$ </div> </div> <div style="text-align: center; margin-top: 20px;"> (iii) $\begin{array}{c} S' \\ / \quad \backslash \\ S \quad Adv \\ \wedge \end{array}$ </div>
Chose TPpt:	if $\alpha = -PAST$, Introduce into \mathbf{Con}_K : $\mathbf{TPpt} := \mathbf{n}$ if $\alpha = +PAST$, Introduce into \mathbf{Con}_K : $\mathbf{TPpt} := \mathbf{o}$ where $\mathbf{o} < \mathbf{n}$ follows from K
Introduce into \mathbf{U}_K:	\mathbf{t}, \mathbf{n}
If $\gamma \neq (i)$ then:	Introduce into \mathbf{Con}_K : $Adv(\mathbf{t})$
<div style="border: 1px solid black; padding: 5px; margin: 5px auto; width: 80%;"> <p style="text-align: center; margin: 0;">If $\delta = -STAT$</p> <p style="margin: 0;">Introduce into \mathbf{U}_K: \mathbf{e}</p> <p style="margin: 0;">Introduce into \mathbf{Con}_K: $\mathbf{e} \subseteq \mathbf{t}$</p> <p style="margin: 0;">If $\beta = past$ then Introduce into \mathbf{Con}_K: $\mathbf{t} < \mathbf{TPpt}$</p> <p style="margin: 0;">If $\beta = fut$ then Introduce into \mathbf{Con}_K: $\mathbf{TPpt} < \mathbf{t}$</p> <p style="margin: 0;">Replace γ with: $\begin{array}{c} S(\mathbf{e}) \\ \wedge \end{array}$</p> </div>	
<div style="border: 1px solid black; padding: 5px; margin: 5px auto; width: 80%;"> <p style="text-align: center; margin: 0;">If $\delta = +STAT$</p> <p style="margin: 0;">Introduce into \mathbf{U}_K: \mathbf{s}</p> <p style="margin: 0;">If $\beta = past$ then Introduce into \mathbf{Con}_K: $\mathbf{s} \circ \mathbf{t}$ Introduce into \mathbf{Con}_K: $\mathbf{t} < \mathbf{TPpt}$</p> <p style="margin: 0;">If $\beta = pres$ then Introduce into \mathbf{Con}_K: $\mathbf{t} \subseteq \mathbf{s}$ Introduce into \mathbf{Con}_K: $\mathbf{TPpt} = \mathbf{t}$</p> <p style="margin: 0;">If $\beta = fut$ then Introduce into \mathbf{Con}_K: $\mathbf{s} \circ \mathbf{t}$ Introduce into \mathbf{Con}_K: $\mathbf{TPpt} < \mathbf{t}$</p> <p style="margin: 0;">Replace γ with: $\begin{array}{c} S(\mathbf{s}) \\ \wedge \end{array}$</p> </div>	
here, \wedge is the 'detensed' tree obtained from the original tree	

Table 2.A: Construction Rule for the top node.

remaining appropriate rules—which are the same as in the previous example—give the following DRS:

e n t $x y$
e \subseteq t
t $<$ n
<i>Peter</i> (x)
<i>room</i> (y)
e : <i>go into</i>(x, y)

Now, we should focus on the second sentence. We need to interpret this sentence against the context already present in the DRS. The important part of the interpretation we are interested in here is the temporal anaphora. This anaphora is handled by using the reference time. The idea is as follows. We need to search an appropriate event in the context to which we can relate the reference time of the sentence that we are just interpreting. The way to find the appropriate event is not always straightforward, but I am going to give some default rules in order to determine it. We focus our attention to either a sequence of sentences in the past tense or a sequence of sentences in the future tense. These rules are as follows. Suppose we are interpreting the sentence S_i of the discourse S_1, \dots, S_n (where $1 < i \leq n$):

- If there is a $j < i$ such that S_j is an event-sentence in the past tense (future tense) and there is no $j < k < i$ such that k is an event-sentence in the past tense (future tense), we stipulate that the reference point be the discourse referent representing the event described by S_j .
- If the previous case is false, and there is a $j < i$ such that S_j is a state-sentence in the past tense (future tense), and there is no $j < k < i$ such that k is a state-sentence in the past tense (future tense), we stipulate that the reference point be the discourse referent representing the location time described by S_j .
- Otherwise, we assume the reference point to be equal to some new event discourse referent.

Once the reference point is determined, we need to establish the relation between the reference point and the eventuality described by the sentence S_i . This relation depends on the feature $STAT$ of S_i . Note that the reference point is always an event, say **t**. This leaves us with two cases open: If $STAT = -STAT$, suppose **e** is the eventuality described by S_i . We should introduce in \mathbf{Con}_K : **t** $<$ **e**. If $STAT = +STAT$, suppose **s** is the eventuality described by S_i . We should introduce in \mathbf{Con}_K : **t** \subseteq **s**.

So, coming back to our example, we have already interpreted the sentence ‘Peter went into the room’ ending up with a DRS K . We suppose given the syntactic structure of the subsequent sentence ‘He turned on the radio’. Thus,

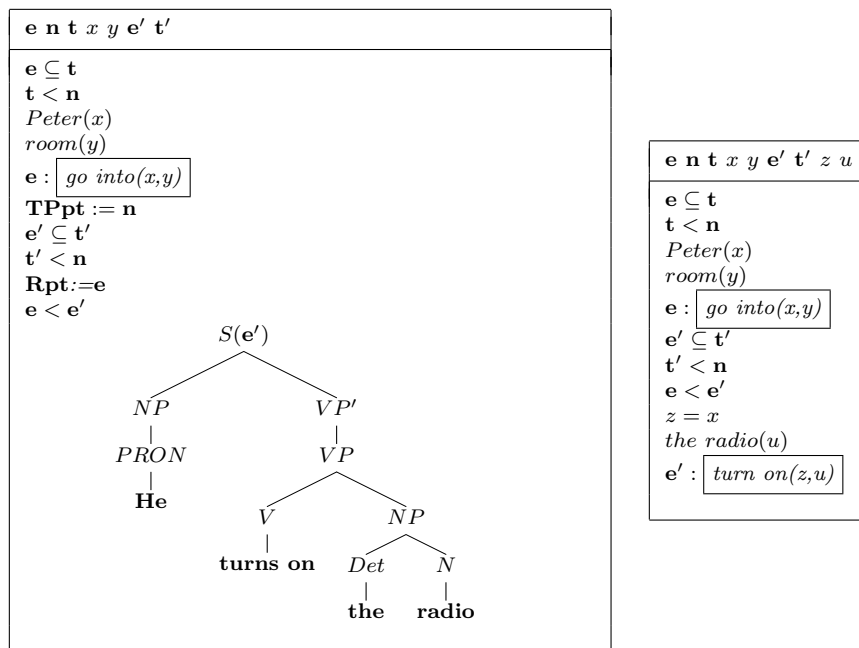


Figure 2.a: Intermediate and final DRS of the second sentence in (2.7).

we need to introduce such syntactic structure in \mathbf{Con}_K . We apply the construction rule for the top node, and next we need to determine the reference point. According to our rules, the referent point is given by the event described by the first sentence, that is, by e . We need to introduce this information and the information of the relation between the reference point and the new eventuality, which is $e < e'$. See figure 2.a.

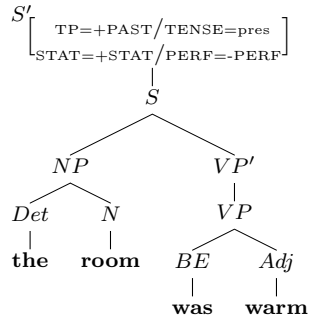
We need here a rule to account for the anaphoric pronoun ‘he’. The description of such a rule will take us too far afield and so I skip it (but see [9, §1.1.1]). Also, the rest of the construction rules applied to the last condition of K do not give us new insight into the temporal anaphora, so I will omit these steps. The final DRS can be seen in figure 2.a.

We turn now to the second example, which is:

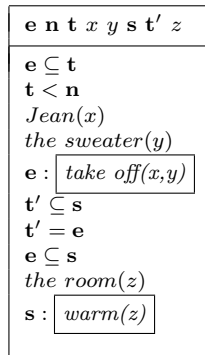
(2.8) Jean took off the sweater. The room was warm.

The by now familiar construction rules applied to this sentence give us a DRS, that I am not going to present here. (It could be read off from the final DRS after representing the second sentence.) It is worth noting that the second sentence is an state-sentence whose temporal perspective is in the past. Thus, it is of

the second kind of structure with regards to the ambiguity I mentioned at the beginning of this section. The syntactic structure of this sentence is:



As can be noted from the construction rule 2.A, when the TP feature has value $+PAST$, we are required to find a temporal perspective point different from \mathbf{n} . DRT does not provide formal rules for such a choice, but, in the case of just two consecutive sentences, we can simply require the \mathbf{TPpt} to be the event described by the first sentence. In the case the first sentence is a state, we can fix the \mathbf{TPpt} as the location time. Since we are only dealing with past tense sentences, the condition of the \mathbf{TPpt} being previous to \mathbf{n} will follow from the axioms on eventualities (see §2.2.1). In our example, we have $\mathbf{TPpt} := \mathbf{e}$, where \mathbf{e} is the event described by the first sentence. At some step in the construction we will need to determine the reference point of this sentence. If we follow the rules of the reference point, we also have that it is \mathbf{e} , and that we should include the condition $\mathbf{e} \subseteq \mathbf{s}$. Note, in the DRS below, that this last condition would have followed in any case. We can apply the familiar construction rules, obtaining:



After this brief summary of the aspects of DRT that I am going to require in Chapter 4, I can turn to a discussion of EC in the next chapter.

Chapter 3

Event Calculus (EC)

The presentation of EC given in this chapter elaborates on that of [19]. However, I will only pay attention to those aspects that are relevant for the translation algorithm that I will develop in the next chapter. In order to do so, I require to do a little bit of elaboration in some points of EC. In particular, I will make explicit: (i) the distinction between different sorts of eventualities; (ii) the codification of real first order predicates by means of Feferman Coding; and (iii) the standard form of each aktionsarten.

An important omission should be mentioned. One of the main elements of EC is the interaction between constraint logic programming and minimal models. I will pay no attention to this, mainly, because of lack of space. Besides, it is not a central theme regarding the translation from DRT in EC, and that is what this part of the thesis is about.

3.1 The Language of EC

EC is a many-sorted first order logic. In the original version, the sort of fluents is considered as a single sort, along with parameterized fluents and fluents for activities. It turns out that, in order to have a neat, general definition of the standard scenario for each aktionsart, we had better distinguish among these different sorts of fluents. Another point I will elaborate from the original version is the use of codes for objects, instead of keeping a separate sort for them. Thus, objects are seen, in the present version, as a finite list of constants of sort $\sigma^{\mathbb{R}}$ (the sort of real numbers, which we already needed for the time parameters).

The language is composed by the following sorts:

1. $\sigma^{\mathbb{R}}$ is the sort of reals,
2. σ^e is the sort of events,
3. σ^f is the sort of non-activity fluents,
4. σ^{act} is the sort of activity fluents.

5. σ^F is the sort of parameterized fluents.

The reader can find a detailed version of the language of EC in appendix 9. It is advisable for the reader to, at least, do a quick reading of such appendix in order to get acquainted with the name of functions and axioms that I am going to use in what follows.

3.2 Eventualities from Predicates

In this section, I will show how to build up an event, a fluent and a fluent for activities, from a real first order predicate. These constructions rely on the Feferman coding (Cf. [3]). Although this coding requires a rather involved machinery, it is a powerful tool, whose utility can be much appreciated later on.

3.2.1 Gödel numbering

A gödel numbering is an assignment of natural numbers (called ‘gödel numbers’) to expressions. Here, expression refers to any term, or formula. Such assignment should meet the following conditions:

1. Different gödel numbers are assigned to different expressions,
2. It is effectively computable what the gödel number of an expression is,
3. It is effectively decidable whether a number is the gödel number of some expression, and, if so, it is effectively computable which expression the number is the gödel number of.

I am not going to give the explicit definition of the assignment of gödel numbers to the expressions of the language of EC. It is enough to note that the procedure in [1, pp.170-1] can be adapted to the language of EC.

Thus, for instance, for each variable x^σ , we have its gödel number, $\ulcorner x^\sigma \urcorner$. Also, if φ is a formula, we have its gödel number, $\ulcorner \varphi \urcorner$.

So we have a way to represent expressions in the language by means of representing its gödel number using the numeral associated with it. For instance, if φ is a formula and $n = \ulcorner \varphi \urcorner$, we represent φ by \bar{n} .

3.2.2 Feferman coding

Let φ be a formula with free variables among $t_1, \dots, t_n, x_1, \dots, x_m$, all of sort $\sigma^{\mathbb{R}}$. Let $\ulcorner \varphi \urcorner$ denote the $\sigma^{\mathbb{R}}$ -numeral of the gödel number of φ . We define¹

$$\varphi[\hat{t}_1, \dots, \hat{t}_n, x_1, \dots, x_m] := \Pi^{m+1}(\ulcorner \varphi \urcorner, x_1, \dots, x_m)$$

Note that $\varphi[\hat{t}_1, \dots, \hat{t}_n, x_1, \dots, x_m]$ is a term of sort $\sigma^{\mathbb{R}}$. Since we are interested in building up eventualities by using this coding, we need functions that return

¹The Π^{m+1} function codifies a $(m+1)$ -tuple into a single real number. For details see appendix §9.

the appropriate sort. This can be carried out by introducing functions I^σ of sort $\sigma^{\mathbb{R}} \rightarrow \sigma$, for $\sigma \in \{\sigma^e, \sigma^f, \sigma^{\text{act}}\}$ and require them to satisfy the following axioms:

1. $I^\sigma(\mathbf{0}^{\sigma^{\mathbb{R}}}) = \mathbf{0}^\sigma$,
2. $I^\sigma \circ (\cdot)' = (\cdot)' \circ I^\sigma$. (where the $(\cdot)'$ in the left-hand side is the successor function of sort $\sigma^{\mathbb{N}}$ and the $(\cdot)'$ in the right-hand side is the successor function of sort σ .)

It is not difficult to see that $\text{EC} \vdash I^\sigma(\bar{n}^{\sigma^{\mathbb{N}}}) = \bar{n}^\sigma$. This will allow us to transport the coded terms from one domain into another, and still keep track of what this code stands for.

3.2.3 Events and fluents with parameters

And now the main result of this section. Let $V(t, x_1, \dots, x_m)$ be a $(m+1)$ -ary predicate of sort $\underbrace{\sigma^{\mathbb{R}} \times \dots \times \sigma^{\mathbb{R}}}_{(m+1)\text{-times}}$. We define:

1. $V[x_1, \dots, x_m]_e := I^{\sigma^e} \circ \Pi^{m+1}(\ulcorner \exists t V \urcorner, \ulcorner x_1 \urcorner, \dots, \ulcorner x_m \urcorner)$,
2. $V[x_1, \dots, x_m]_f := I^{\sigma^f}(V[\hat{t}, x_1, \dots, x_m])$,
3. $\bar{V}[x_1, \dots, x_m]_f := I^{\sigma^f} \circ \Pi^{m+1}(\ulcorner \neg V \urcorner, \ulcorner x_1 \urcorner, \dots, \ulcorner x_m \urcorner)$,
4. $V[x_1, \dots, x_m]_{\text{act}} := I^{\sigma^{\text{act}}}(V[\hat{t}, x_1, \dots, x_m])$,

Note that these are *terms*, not formulas. They represent a numeral, and we will have to use a process of decoding in order to determine which variables and predicate they are built up from.

This process can be extended in a straightforward way to any formula φ whose free variables are among $t_1, \dots, t_n, x_1, \dots, x_m$.

3.3 Constraint Logic Programming

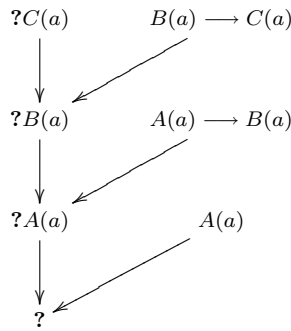
Suppose $A(x), B(x)$, and $C(x)$ are predicates and a is an object. From the set of premises

$$\{A(a) \rightarrow B(a), B(a) \rightarrow C(a), A(a)\}$$

and the modus ponens rule we can deduce the formula $C(a)$: $A(a)$ and $A(a) \rightarrow B(a)$ give $B(a)$. This and $B(a) \rightarrow C(a)$ give $C(a)$. This is a derivation of $C(a)$ from the set of premises. In constraint logic programming, on the other hand, we work backwards. We start from the set of premises Γ and a formula φ , and we ask whether φ can be derived from Γ . So, we have a query, denoted by $?\varphi$, and we try to derive the query from the premisses. We will illustrate this procedure by means of an example. Suppose Γ is $\{A(a) \rightarrow B(a), B(a) \rightarrow C(a), A(a)\}$ and φ is $C(a)$. The derivation of the query proceeds as follows:

1. Ask the query: $?C(a)$.
2. Search into the set of premisses a premise whose consequent is $C(a)$.
3. If such a premise exists: Replace $C(a)$ in the query by the antecedent of this premise. Go back to step 1 with the new query.
4. If no such a premise exists: The derivation ends in a failure.
5. If at some step the query is empty, the derivation stops in a success.

We can represent the derivation in a graphical way as follows:



If we follow the arrows in the inverse direction we can get the derivation of $C(a)$ out of Γ by means of modus ponens. So far so good, but this is only the first step. The derivations in which we are interested include universally quantified premisses. This implies that we will need to deal with *unification* and with constraints of the form $\tau_1 = \tau_2$, where τ_1 and τ_2 are terms. Furthermore, we should allow for the possibility of there being queries containing more than one clause. We are going to handle with both improvements at the same time in the following example. Suppose that our set of premisses is

$$\Gamma = \{\forall x(A(x) \rightarrow B(x)), \forall y((B(y) \wedge D(y)) \rightarrow C(y)), A(a), D(a)\}$$

We make the question whether we can derive $C(a)$ out of this premisses. In order to give a neat procedure, we need the following clarifications:

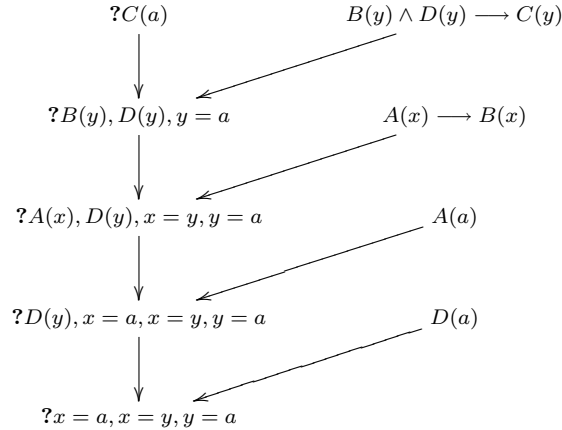
- We note that the premisses we will be dealing with are all in prenex normal form. That is, they take on the form $Q_1 \dots Q_n \psi$ where Q_i is $\forall x_i$ for an appropriate variable x_i , and ψ is a quantifier free formula. Hereafter we will leave out the quantifiers and deal only with ψ . The process of unification will allow the derivation to behave as though we had the quantifiers.
- We say that $\varphi[\tau]$ can be *unified* to $\psi(x)$ if the formula² $[\tau/x]\psi$ is equal to $\varphi[\tau]$. The *constraint* of the unification is $x = \tau$. For instance, $C(a)$ can be unified to $C(x)$ with the constrain $x = a$.

²Here $[\tau/x]\psi$ is the formula $\psi(x)$ where all the free occurrences of x have been replaced with τ .

With this two points in mind it is not difficult to explain the procedure of the derivation as follows:

1. Ask the query: $?C(a)$.
2. Search into the set of premisses a premise whose consequent can be unified to $C(a)$.
3. If such a premise exists: Replace $C(a)$ in the query by the antecedent of this premise plus the constraint that allowed the unification. Go back to step 1 with the new query.
4. If no such a premise exists: The derivation ends in a failure.
5. If at some step the query contains only constraints, the derivation stops in a success.

These steps applied to our example can be represented graphically as follows:



Thus, whenever the constraints $x = a$ and $y = a$ are true, we can derive $C(a)$ out of Γ by means of modus ponens and the logical axioms of First-Order Logic—the axiom of universal instantiation is of paramount importance here.

So far we have been dealing with positive clauses, i.e., none of the formulas contained negated sub-formulas. However, we want to allow the possibility of premisses having negated clauses. EC and its version of constraint logic programming deals with negated clauses by means of a procedure called *negation as failure*. A short description of it is the following. Whenever we have a clause of the form $\neg\varphi$ in the query we can start a new query of the form $?\varphi$. If such a query fails, then we will leave out the clause $\neg\varphi$ from the original query.

3.4 Integrity Constraints

We turn now to an important element in EC: integrity constraints. The intuitive idea of an integrity constraint (IC hereafter) is that it requires a particular

update of a ‘database’. In our case, this database is a *scenario*. A scenario is, intuitively, the description of the initial conditions of a model. We are not going into details here (but see [19, p. 43]). Again, we will work with an example. Suppose our scenario is

$$\{\text{INITIATES}(\textit{break}, \textit{being_broken}, t)\}$$

where *break* is an event, *begin_broken* a fluent, and *t* a real variable (representing time).³ An IC takes on the form of a query plus a requirement of success or failure. If the requirement is that of success (failure), the scenario should be updated in such a way that the query succeeds (fails). There are two constraints on the possible formulas that we can use to update the scenario. In the first place, the new formula should be consistent with the scenario, and second, the update should be non trivial. If we cannot find an appropriate formula that we can use to update the scenario, we say that the IC *fails*. Let us develop an example. Suppose our IC is

$$?\text{HOLDSAT}(\textit{being_broken}, 10) \text{ SUCCEEDS}$$

The most immediate way to update the scenario in order for the query

$$?\text{HOLDSAT}(\textit{being_broken}, 10)$$

to hold is to introduce the sentence $\text{HOLDSAT}(\textit{being_broken}, 10)$ itself in the scenario, yielding:

$$\{\text{INITIATES}(\textit{break}, \textit{being_broken}, t), \text{HOLDSAT}(\textit{being_broken}, 10)\}$$

This scenario can be shown to be consistent, so the first constraint is not broken. Furthermore, the query $?\text{HOLDSAT}(\textit{being_broken}, 10)$ clearly succeeds. However, the second constraint is broken. The update is trivial. A non trivial update consists of the introduction in the scenario of (i) the antecedent of an axiom of EC, or (ii) the antecedent of a conditional formula⁴ already present in the scenario. In our example, we have two possible non trivial updates. We can introduce $\text{INITIALLY}(\textit{being_broken})$, or we can introduce $\text{HAPPENS}(\textit{break}, t') \wedge t' < 10$.

The convention to chose among the possible updates before an IC of the form

$$?\text{HOLDSAT}(f, t) \text{ SUCCEEDS}$$

is as follows:

- First, try to update the scenario with $\text{HAPPENS}(e, t)$ for an appropriate value of *e*, and appropriate inequalities in *t*. If this does not work, try the following:
- Update the scenario with $\text{INITIALLY}(f)$. It is not difficult to see that this last clause always leads to the success of the IC.

³Note that the sentence *is* of the form $\forall t \text{INITIATES}(\textit{break}, \textit{being_broken}, t)$, but, as explained before, we omit the universal quantifier.

⁴A conditional formula is something like $\varphi \rightarrow \psi$.

The convention to chose among the possible updates before an IC of the form

$$?HAPPENS(e, t) \text{ SUCCEEDS}$$

is as follows:

- First, try to update the scenario with antecedents of conditional formulas in the scenario. If this does not work, try the following:
- Update the scenario with the formula $HAPPENS(e, t)$. and the appropriate value of e and constraints on t . It is not difficult to see that this last clause always leads to the success of the IC.⁵

3.5 Aktionsart

Eventualities, as defined in section §2.2, is a common name for events and states. EC differs in its ontology from that of DRT since it uses fluents, which are more complex entities than states, given their different roles in causality (they can represent a state but also a complex event which develops through time, and also actions, which are able to cause a continuous change on the parameterized fluents). However, all of those are eventualities, in the sense of being ‘temporal objects’. This term will receive a formal treatment in EC with the aim of capturing, within a single element, all possible aspectual properties of eventualities.

Let f be a fluent of sort σ^f , \mathbf{f} a fluent of sort σ^{act} , and F a fluent of sort σ^F . The formal definition of an eventuality is that of a structure (\mathbf{f}, F, e, f) , where:

1. \mathbf{f} is a fluent which represents an activity, something which exerts a force,
2. F is a parameterized fluent, representing a parameterized object or state, which is driven by the force \mathbf{f} ,
3. e is the culminating event, representing a canonical goal,
4. f is a fluent which represents the state of having achieved the goal. Also, it can be just a state of affairs.

EC acknowledge six types of aspectual properties of VP: state, activity (strict), activity (wide), accomplishment, achievement, and points. Each category has a different representation as an eventuality. Remember that an eventuality is a quadruple, i.e., we have four slots to fill in. However, not every aspectual property requires all of the slots to be filled in. In fact, requiring some slots and no others to be necessarily filled in is the way EC defines the aspectual properties. The requirements for the five aspectual properties aforementioned are summarized in the following table:

⁵This is also a trivial update, but the fact is that there is no other way to update the scenario so as to make the query successful.

Aktionsart	To fill in
State	$\langle -, -, -, + \rangle$
Activity (strict)	$\langle +, -, -, - \rangle$
Activity (wide)	$\langle +, +, -, - \rangle$
Accomplishment	$\langle +, +, +, + \rangle$
Achievement	$\langle -, -, +, + \rangle$
Point	$\langle -, -, +, - \rangle$

Each aspectual property has a default representation in a scenario. The construction of the scenario from a VP and its aktionsart will be of paramount importance afterwards. We shall try to make explicit the steps of the general case in this construction. From now on we suppose given a VP in the form of a $(m + 1)$ -ary First-Order Predicate $V(t, x_1, \dots, x_m)$, where x_1, \dots, x_m and t are variables of sort $\sigma^{\mathbb{R}}$. In the following subsections we will deal with each aktionsart in turn. (See section §3.2.3 for details about the construction of an event, a fluent of sort σ^f and a fluent of sort σ^{act} from V .)

States If the VP is to be a state, we use the fluent $V[x_1, \dots, x_m]_f$. The cases we are interested in here do not involve the use of a scenario for states. Nevertheless, the causal behavior of states is constrained due to the axioms of EC. Note that, mainly, states cannot appear in the first component of the TRAJECTORY predicate, which means that states are inert for causation —the states do not cause a continuous change on parameterized fluents.

Activities in the strict sense If the VP is to be a strict activity, we use the fluent $V[x_1, \dots, x_m]_{act}$. The activities in the strict sense do not require a scenario either. The constraints in its causal behavior are given by EC axioms. Note that, mainly, activities are the only kind of fluents appearing in the first component of the TRAJECTORY predicate, and they do not have a predicate RELEASES —whereas events and parameterized fluents do.

Activities in the wide sense If the VP is to be a wide activity, we use the fluent $V[x_1, \dots, x_m]_{act}$, a parameterized fluent F of sort σ^F , and an injective real function $g(r)$ of sort $\sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$. The scenario for this aktionsart should be:

$$\{\text{HOLDSAT}(F, g(y), t) \rightarrow \text{TRAJECTORY}(V[x_1, \dots, x_m]_{act}, t, F, g(y + d), d)\}$$

Accomplishments If the VP is to be an accomplishment, we introduce the fluent $V[x_1, \dots, x_m]_{act}$, a parameterized fluent F , events *start* and *finish*, constants a and c of sort $\sigma^{\mathbb{R}}$, and a monotone increasing function $g(r)$ of sort $\sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$. As can be read off from the formulas that are going to be introduced in the scenario (see below), the intended meaning of *start* and *finish* is that of the events that trigger and finish (respectively) the fluent $V[x_1, \dots, x_m]_{act}$. In a minimal model, the interpretation of $V[x_1, \dots, x_m]_{act}$ would be a semiopen interval $(r, s]$, where r is such that $\text{HAPPENS}(\textit{start}, r)$ holds, and s is such that $\text{HAPPENS}(\textit{finish}, s)$ holds. Besides, a and c are intended to be the initial and

the final state (respectively) of the parameterized fluent F . Note that, when F reaches the value c , it triggers the occurrence of the *finish* event.

With this ingredients we build up the following scenario:

$$\begin{aligned} &\{a < c, \text{HOLDSAT}(F, y, t) \wedge \text{HOLDSAT}(F, z, t) \rightarrow y = z, \\ &\text{INITIALLY}(F, a), \text{INITIATES}(start, V[x_1, \dots, x_m]_{act}, t), \\ &\text{TERMINATES}(finish, V[x_1, \dots, x_m]_{act}, t), \\ &\text{HOLDSAT}(V[x_1, \dots, x_m]_{act}, t) \wedge \text{HOLDSAT}(F, c, t) \rightarrow \text{HAPPENS}(finish, t), \\ &\text{HOLDSAT}(F, g(y), t) \rightarrow \text{TRAJECTORY}(V[x_1, \dots, x_m]_{act}, t, F, g(y + d), d), \\ &\text{RELEASES}(start, F, y, t)\} \end{aligned}$$

Achievements If the VP is to be an achievement, we introduce the event $V[x_1, \dots, x_m]_e$, and fluents $C[x_1, \dots, x_m]_f$ and $\overline{C}[x_1, \dots, x_m]_f$, which come from coding a predicate $C(t, x_1, \dots, x_m)$. The scenario for this aktionsart is:

$$\{\text{HOLDSAT}(C[x_1, \dots, x_m]_f, t) \rightarrow \text{INITIATES}(V[x_1, \dots, x_m]_e, t, \overline{C}[x_1, \dots, x_m]_f)\}$$

Points If the VP is to be a point, we introduce the event $V[x_1, \dots, x_m]_e$. No scenario is required.

3.6 IC's for Tensed Sentences

The semantic representation in EC of a sentence comes down to the construction of a scenario —depending on its aktionsart— and an integrity constraint —depending on both its aktionsart and its tense. In the next section we are going to give an almost complete example of the semantic representation carried out in EC. But before doing that, and because we will need it later on for some details in the translation algorithm, we are going to make clear how to give an IC for sentences which are either achievements or accomplishments, and which are in the past tense. The interesting part here is the temporal anaphora, and the appropriate link between the parameterized fluent in an accomplishment's scenario and its IC.

To begin with, suppose given a VP which has been determined to be an achievement, and suppose it comes from a sentence S in past tense. The IC for S is:

$$?\text{HOLDSAT}(f_1, R), \dots, \text{HOLDSAT}(f_n, R), \text{HAPPENS}(e, R), R < now \text{ SUCCEEDS}$$

where e is the event given by the scenario of the VP (say $V[x_1, \dots, x_m]_e$), f_n is the fluent given by the scenario (say $C[x_1, \dots, x_m]_f$), and f_1, \dots, f_{n-1} are context fluents. If we want a formal translation from a linguistic input, we should specify some rules to fix such context fluents. Since the semantic representation is intended to act upon a narrative discourse, we assume given a sequence of sentences S_1, \dots, S_k, S . Thus, we are going to search the context

fluents in such sentences. Let VP_i , Akt_i , and $Tense_i$ be the verb phrase, the aktionsart, and the tense of sentence S_i , respectively ($i = 1, \dots, k$). The rules we are going to follow here are:

- If $Tense_i$ is past, and Akt_i is state, then f_i is the fluent represented by VP_i .
- If $Tense_i$ is past, and Akt_i is accomplishment, then f_i is the fluent $F[c]$, represented by the scenario of VP_i .
- If $Tense_i$ is past, and Akt_i is achievement, then f_i is the fluent $\overline{C}[x_1, \dots, x_m]_f$, represented by the scenario of VP_i .

This is enough to handle with the cases we are interested in here.

3.7 Semantic Representation in EC

Here we will give the representation in EC of the mini-discourse:

(3.1) Jean took off his sweater. The room was warm.

Recall that the purpose of this part of the project is to define a translation algorithm from a DRS into a semantic representation in EC. But, before we can get into details of the translation, it is convenient to make clear what the semantic representation in EC looks like. We will make the semantic representation of (3.1) from scratch —i.e. intuitively— in this section in order to state clearly what precisely the end point of the translation algorithm is. Since we already analyzed (3.1) in DRT, the setup for stating the translation algorithm will be completed, for we will have its starting and end points defined.

It should be noted that we are representing both sentences simultaneously, although we could have done this in a sequential way. The adaptation to this is straightforward.

1. *The first step in the representation consists in the identification of the variables and constants of the language with which we represent the important elements of the sentences.* The objects referred to by the sentences are ‘Jean’, ‘the sweater’, and ‘the room’. In order to represent them we need predicates of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$: $Jean(t, x)$, $the\ sweater(t, y)$, and $the\ room(t, z)$, respectively. Then, we build up fluents for each predicate: $Jean[x]_f$, $the\ sweater[y]_f$, and $the\ room[z]_f$, respectively. At first sight this could look odd, but this turns out to be a wise decision because DRT also codifies objects as predicates, and then the translation will be more smooth. However, we will also assume that we have constants **Jean**, **the sweater**, and **the room** of sort $\sigma^{\mathbb{R}}$.

The eventualities in the sentences are ‘take off’ and ‘be warm’. They will yield predicates $take\ off(t, x, y)$ and $warm(t, x)$ of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$ and $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$, respectively. Now, we need to attach to each of them its

respective aktionsart. Thus, since ‘take off’ is an achievement, we need an event and two (complementary) fluents. The event is $take\ off[x, y]_e$. The fluents are derived from the predicate $on(t, x, y)$ of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$, and are $on[x, y]_f$ and $\overline{on}[x, y]_f$. Finally, since ‘be warm’ is a state, we shall represent it by means of the fluent $warm[z]_f$.

2. *The second step consists in the construction of the scenario using the constraints on each eventuality.* We first represent the achievement ‘take off’: the constraint on the achievement ‘take off’ gives the scenario:

$$\{\text{HOLDSAT}(on[x, y]_f, t) \rightarrow \text{INITIATES}(take\ off[x, y]_e, \overline{on}[x, y]_f)\} \quad (3.2)$$

Now, as for the state ‘being warm’: the states don’t determine a scenario but in special cases, and this is not one of those. So the scenario is just (3.2).

3. *The third step is the construction of the IC for each sentence.* We need to state two kinds of information in the IC. The first is the temporal information of the sentence, and the second is the existence of the objects referred to by the sentence, and which appear as parameters of the eventuality. This is carried out for the sentence ‘Jean took off the sweater’ in the following IC:

$$\begin{aligned} & ?\text{HOLDSAT}(Jean[x]_f, R_1), \text{HOLDSAT}(the\ sweater[y]_f, R_1) \\ & \text{HOLDSAT}(on[x, y]_f, R_1), \text{HAPPENS}(take\ off[x, y]_e, R_1), \quad (3.3) \\ & R_1 < now, x = \text{Jean}, y = \text{the sweater} \text{ SUCCEEDS} \end{aligned}$$

The same goes for the sentence ‘The room was warm’, yielding the IC:

$$\begin{aligned} & ?\text{HOLDSAT}(the\ room[z]_f, R_2), \\ & \text{HOLDSAT}(warm[z]_f, R_2), R_2 < now, \quad (3.4) \\ & z = \text{the room} \text{ SUCCEEDS} \end{aligned}$$

4. *The final step is the verification of the IC’s and the update of the scenario.* We verify first the IC in (3.3). According to the rules of an IC for events and fluents in page 34 we update the scenario with the respective HAPPENS, INITIALLY, and constraint formulas, yielding:

$$\begin{aligned} & \{\text{HOLDSAT}(on[x, y]_f, t) \rightarrow \text{INITIATES}(take\ off[x, y]_e, \overline{on}[x, y]_f), \\ & \text{INITIALLY}(Jean[x]_f), \text{INITIALLY}(the\ sweater[y]_f), \\ & \text{HAPPENS}(take\ off[x, y]_e, R_1), \text{INITIALLY}(on[x, y]_f, R_1 < now)\} \end{aligned}$$

We continue with the verification of the IC in (3.4). According to the rules of an IC for fluents in page 33 we need to update the scenario with

formulas INITIALLY. This gives us the following update:

$$\begin{aligned} &\{\text{HOLDSAT}(\text{on}[x, y]_f, t) \rightarrow \text{INITIATES}(\text{take of } f[x, y]_e, \overline{\text{on}}[x, y]_f), \\ &\quad \text{INITIALLY}(\text{Jean}[x]_f), \text{INITIALLY}(\text{the sweater}[y]_f), \\ &\quad \text{HAPPENS}(\text{take of } f[x, y]_e, R_1), \text{INITIALLY}(\text{on}[x, y]_f), R_1 < \text{now}, \\ &\quad \text{INITIALLY}(\text{the room}[z]_f), \text{INITIALLY}(\text{warm}[z]_f), R_2 < \text{now}\} \end{aligned}$$

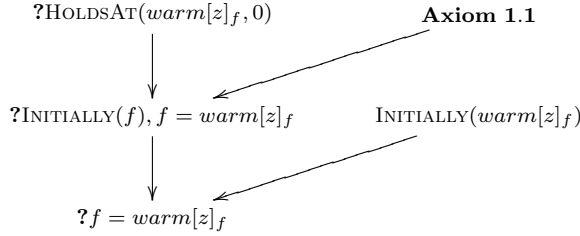
Let us verify that the information we intended example (3.1) to represent is given in the final scenario. This information should be true in any minimal model of the scenario. The verification will take the form of a list of propositions which should be true or false at certain intervals of time. The list we have in mind is:

1. $\text{HOLDSAT}(\text{Jean}[x]_f, t)$, $\text{HOLDSAT}(\text{the sweater}[y]_f, t)$,
 $\text{HOLDSAT}(\text{the room}[z]_f, t)$, and $\text{HOLDSAT}(\text{warm}[z]_f, t)$ are true for $t \geq 0$.
2. $\text{HOLDSAT}(\text{on}[x, y]_f, t)$ is true for $0 \leq t \leq R_1$, and false for $t > R_1$.
3. $\text{HOLDSAT}(\overline{\text{on}}[x, y]_f, t)$ is false for $0 \leq t \leq R_1$, and true for $t > R_1$.

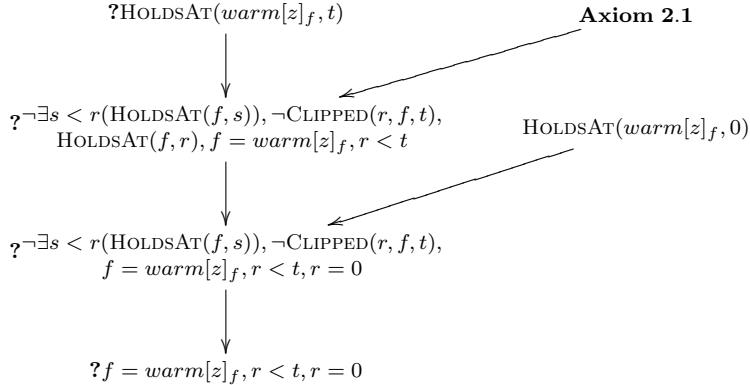
We proceed to prove some of these claims.

Claim 1. $\text{HOLDSAT}(\text{warm}[z]_f, t)$ is true for every $t \geq 0$.

Proof. The following is a derivation of $\text{HOLDSAT}(\text{warm}[z]_f, 0)$ from the scenario and the axioms of EC:



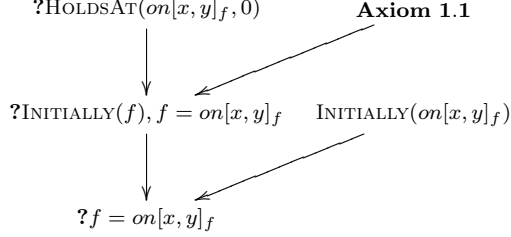
Using this we can derive that $\text{HOLDSAT}(\text{warm}[z]_f, t)$ for any arbitrary $t > 0$:



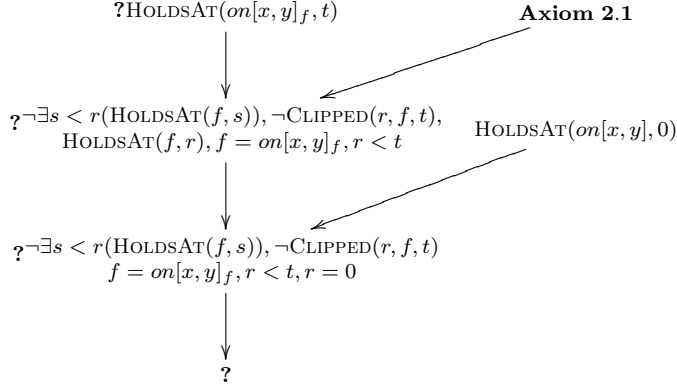
The last step in the derivation follows from negation as failure applied to $\neg\exists s < r(\text{HOLDSAT}(f, s))$ and to $\neg\text{CLIPPED}(r, f, t)$ along with the constraints $r = 0$, $r < t$, and $f = \text{warm}[z]_f$. \square

Claim 2. $\text{HOLDSAT}(on[x, y]_f, t)$ is true for every $t \leq R_1$.

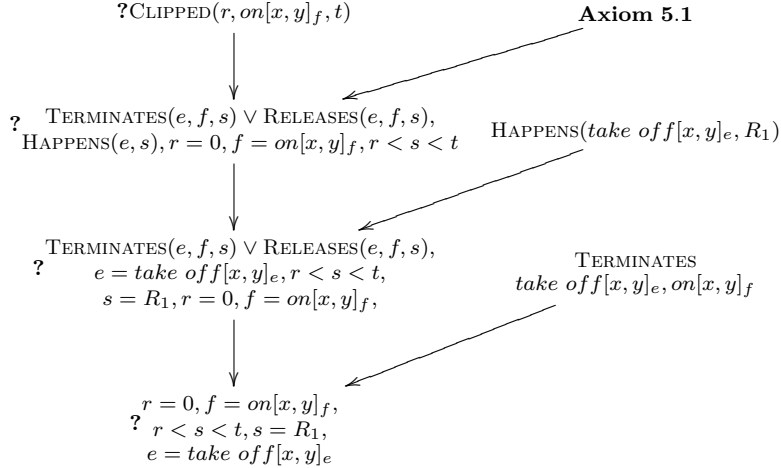
Proof. From the scenario and EC axioms we can derive that $\text{HOLDSAT}(on[x, y]_f, 0)$:



Using this we can derive that $\text{HOLDSAT}(on[x, y]_f, t)$ for any $t \leq R_1$:



The last step in the derivation follows from negation as failure applied to $\neg\exists s < r(\text{HOLDSAT}(f, s))$ and to $\neg\text{CLIPPED}(r, f, t)$ along with the constraints $r = 0$, $f = on(\textit{Jean, the sweater})$, and $t \leq R_1$. It is worth mentioning that without the latter we could not have failed in the derivation of $\text{CLIPPED}(r, f, t)$. For instance, for any $t > R_1$ it follows that $\text{CLIPPED}(r, f, t)$. Here is the proof of this fact—that $\text{CLIPPED}(r, f, t)$ is true for $t > R_1$:



□

Claim 3. $\text{HOLDSAT}(on[x, y]_f, t)$ is false for every $t > R_1$.

Proof. It is not difficult to read from the constraints $r < s < t, s = R_1$ in the previous derivation that whenever $r < R_1 < t$ then $\text{CLIPPED}(r, on[x, y]_f, t)$ is true. Therefore, the derivation of $\text{HOLDSAT}(on[x, y]_f, t)$ cannot be done via axioms 2.1 or 3.1. It is not difficult to see that such derivation is not possible via axiom 1 either. This means that no derivation of this claim in the scenario is possible. Therefore, the negation of it is true due to our definition of negation as failure. □

Chapter 4

Translating from DRS

4.1 Translation of an Example

The example of a semantic representation given in the previous chapter can be considered as the outcome of a translation algorithm. As I said before, the idea of the first part of the thesis is to come up with a formalized way to go from DRT to EC. In order to do so, I will explore a way of going from the DRS previously built in §2.3.2 to a semantic representation in EC as close as possible to the representation just made in the previous chapter.

The first inconvenience in making the step from DRT to EC is that the latter depends very much on the aktionsart of the VP. For instance, the definition of the scenario depends explicitly on the aktionsart of the VP. Thus, such information should be available at some step of the representation. The problem is that this information is not present in the DRS, save in the form of the distinction between events and states. I will address this problem later on.

In the previous section I identified four steps in the semantic representation. The fourth step was just a verification, or actualization brought about by the set up —set up consisting of scenario plus IC. Thus, a translation has only to do with the first three steps of the representation, namely, the identification of the language, the construction of the scenario, and the construction of the IC's.

4.1.1 Identification of the language

The first step is the identification of the language constants. The algorithm receives a DRS and scans each of its conditions. We have a table of instructions triggered by DRS conditions (see list of instructions (4.1) below). Any time a DRS condition has the form of one of the conditions in the table, the respective instruction in the table is carried out. The outcome of the carrying out of an instruction is the adding of information in a new structure that I will call *Intermediate Information*, or *II*.

The list of instructions is:

If DRS Condition is: Follow instruction:

(4.1)	$P(x)$	<ul style="list-style-type: none"> (i) Create a predicate $P(t, x)$ of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$. (ii) Introduce in \mathcal{II} fluent $P[x]_f$. (iii) Create a constant P of sort $\sigma^{\mathbb{R}}$. (iv) Introduce in \mathcal{II} line '$x = P$'.
-------	--------	--

$e : \boxed{V(x_1, \dots, x_n)}$		<ul style="list-style-type: none"> (i) Create a predicate $V(t, x_1, \dots, x_n)$ of sort $\underbrace{\sigma^{\mathbb{R}} \times \dots \times \sigma^{\mathbb{R}}}_{(n+1)\text{-times}}$. (ii) Introduce in \mathcal{II} '$V(t, x_1, \dots, x_n)$ comes from e'.
----------------------------------	--	---

Let us work with our example. For ease of presentation I will reproduce the DRS of example (3.1) here:

e n t x y s t' z
e \subseteq t
t $<$ n
<i>Jean(x)</i>
<i>the sweater(y)</i>
e : $\boxed{\textit{take off}(x,y)}$
t' \subseteq s
t' = e
e \subseteq s
<i>the room(z)</i>
s : $\boxed{\textit{warm}(z)}$

The idea is to examine each condition of the DRS in turn. If the condition is of one of the two forms in list (4.1), we follow the respective instructions. In the example, the first and second conditions in the DRS are of a different form from the conditions in the list. Once we scan the third condition, we find that it has the form of the first condition of the list (4.1). Accordingly, we should introduce in \mathcal{II} a new predicate and derive a fluent from it. The same goes for conditions fourth and ninth. Once we scan the fifth, we find that it has the form of the second triggering condition. We should introduce in \mathcal{II} an appropriate predicate. The same goes for the tenth line. The application of this step returns the following \mathcal{II} :

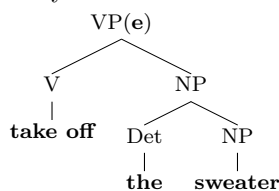
\mathcal{II}
<i>Jean</i> [x] _{<i>f</i>} $x = \mathbf{Jean}$ <i>the sweater</i> [y] _{<i>f</i>} $y = \mathbf{the\ sweater}$ <i>the room</i> [z] _{<i>f</i>} $z = \mathbf{the\ room}$ <i>take off</i> (t, x, y) comes from \mathbf{e} <i>warm</i> (t, z) comes from \mathbf{s}

4.1.2 Construction of the scenario

Now, we should assume that there is a database entry of the form:

$$\begin{aligned} \lambda_1 &\mapsto \langle s_1^1, s_2^1, s_3^1, s_4^1 \rangle \\ \lambda_2 &\mapsto \langle s_1^2, s_2^2, s_3^2, s_4^2 \rangle \\ &\vdots \\ \lambda_k &\mapsto \langle s_1^k, s_2^k, s_3^k, s_4^k \rangle \\ &\vdots \end{aligned}$$

where λ_k is a DRT eventuality, and $s_i^k \in \{+, -\}$ for $i = 1, \dots, 4$, and $k = 1, \dots$. This database will be in charge to store the aspectual information of the VP's. This is the trickiest part of the whole algorithm. Since the aspectual information is a property of, usually, the VP, then we need to associate the respective aktionsart to the VP when we have access to both verb and direct object (and complements). Therefore, this database is assumed to be constructed *while the DRS is being constructed*. Let us explain this by using our example. The construction of the DRS from the sentence 'Jean took of the sweater' has to go through the step where we analyze the reducible condition represented by:



At this stage we have access to the full VP, and we can assume a way to attach an appropriate aktionsart to it. Since we also have the information of the eventuality by means of which it is represented (\mathbf{e} in this case), we can attach to \mathbf{e} an aktionsart: $\mathbf{e} \mapsto \langle -, -, +, + \rangle$.

In order to handle with our example, we assume the following database:

$$\begin{aligned} \mathbf{e} &\mapsto \langle -, -, +, + \rangle \\ \mathbf{s} &\mapsto \langle -, -, -, + \rangle \end{aligned}$$

We need more instructions in order to attach an aktionsart to each predicate in \mathcal{II} . These instructions act on the already built \mathcal{II} . We go line by line in \mathcal{II}

and determine whether they are predicates. If so, we look up in the database which aktionsart is attached to it.

Following this procedure in our example we end up with the following \mathcal{II} :

\mathcal{II}
<i>Jean</i> [x] _{<i>f</i>} $x = \mathbf{Jean}$ <i>the sweater</i> [y] _{<i>f</i>} $y = \mathbf{the\ sweater}$ <i>the room</i> [z] _{<i>f</i>} $z = \mathbf{the\ room}$ <i>take of f</i> (t, x, y) comes from e is $\langle -, -, +, + \rangle$ <i>warm</i> (t, z) comes from s is $\langle -, -, -, + \rangle$

We are now in a position to create an scenario using \mathcal{II} . We can just take each eventuality in \mathcal{II} and its aktionsart, and create a scenario for it in accordance with what was said in §3.5. Finally, we will unite all of these scenarios in a single big scenario.

We put the first two steps —i.e. database and many scenarios—in the following instructions:

	If line is a 1stOL predicate	Follow instruction
$V(x_1, \dots, x_n, t)$ comes from λ	(i) Look up in the database for the aktionsart of λ , and add the information 'is $\langle s_1, s_2, s_3, s_4 \rangle$ ' to the respective line in \mathcal{II} .	
(4.2)	(ii) Create a new scenario for V in accordance with its aktionsart (Cf. §3.5).	

After these instructions are carried out in \mathcal{II} we need to execute instruction: Unite all the previously created scenarios into a single one.

Accordingly, and continuing with our example, we end up with the following scenario:

$$\{\text{HOLDSAT}(\text{on}[x, y]_f, t) \rightarrow \text{INITIATES}(\text{take of f}[x, y]_e, \overline{\text{on}}[x, y]_f)\}$$

4.1.3 Construction of the IC's

The creation of the IC should serve three purposes: (i) we require that the fluents representing the objects hold; (ii) we need to codify the temporal structure of the sentence; (iii) we need to fit together the information in (i) and (ii).

The first task is easily accomplished by the following instructions —again based on a scan of the lines of \mathcal{II} :

	If line is	Follow instruction
(4.3)	a fluent f	Introduce in \mathcal{II} : $?HOLDSAT(f, R)$ SUCCEEDS.
	$x = P$	Introduce in \mathcal{II} : $?x = P$ SUCCEEDS

Further, we need the temporal information, codified in the DRS as a list of appropriate binary relations between eventualities (for details, see below, and table 4.B). We need to go over the lines of \mathcal{II} , looking for predicates. When we find one, we should record from which DRT-eventuality it comes, and then we go back to the DRS and create the appropriate list of relations. The way of creating this list can proceed along the following lines:

	If line is a predicate	Follow instructions
(4.4)	$V(t, x_1, \dots, x_n)$ comes from λ	<p>(i) Find in the original DRS all the conditions of the form $\lambda \subseteq \kappa$, $\kappa \subseteq \lambda$, or $\lambda \circ \kappa$, determining thus a finite list $\kappa_1, \dots, \kappa_k$ of eventualities related with λ.</p> <p>(ii) For each κ_i ($i = 1, \dots, k$) find in the DRS all the conditions of the form $\kappa_i < \gamma$, $\gamma < \kappa_i$, or $\kappa_i = \gamma$. Collect those γ's and make a finite list $\kappa_{i,1}, \dots, \kappa_{i,r_i}$ of eventualities related with κ_i (i.e., the γ's are referred to as $\kappa_{i,j}$). Do this once again obtaining a third list $\kappa_{i,j,1}, \dots, \kappa_{i,j,s_j}$.</p> <p>(iii) According to lemma 1, this information is enough to determine the tense of the eventuality.</p>

The tense of the eventuality and its aktionsart are enough to find the appropriate IC according to the instructions in [19, ch. 8].

Now, we turn to the proof of the important lemma that the tense of the eventuality can be read off from the DRS.

Lemma 1. *Suppose the DRS K was built up following the construction rule (2.A) and the rules for the reference time described in chapter 2. Suppose K contains a condition of the form*

$$\lambda : \boxed{V(x_1, \dots, x_n)}$$

Let $\kappa_1, \dots, \kappa_n$ be eventualities such that $\lambda \subseteq \kappa_i$, $\kappa_i \subseteq \lambda$, or $\lambda \circ \kappa_i$ are conditions of K (call this kind of conditions C_i^1). Further, let $\kappa_{i,1}, \dots, \kappa_{i,r_i}$ be eventualities

TENSE TABLE			
Branch of conditions	$\lambda \subseteq \kappa_i$ $\kappa_i < \kappa_{i,j}$ $\kappa_i \subseteq \lambda$ $\kappa_i = \kappa_{i,j} (\neq \mathbf{n})$	$\lambda \circ \kappa_i$ $\kappa_i < \kappa_{i,j}$ $\kappa_i \subseteq \lambda$ $\kappa_i = \mathbf{n}$	$\lambda \subseteq \kappa_i$ $\kappa_{i,j} < \kappa_i$ $\lambda \circ \kappa_i$ $\kappa_{i,j} < \kappa_i$
Tense	Past	Present	Future

Table 4.A: This table shows the tense attached to each branch.

such that $\kappa_i < \kappa_{i,j}$, $\kappa_{i,j} < \kappa_i$, or $\kappa_i = \kappa_{i,j}$ are conditions in K (for $j = 1, \dots, r_i$, and $i = 1, \dots, n$) (call this kind of conditions $C_{i,j}^2$). Call a pair $\begin{matrix} C_i^1 \\ C_{i,j}^2 \end{matrix}$ of conditions a **branch**.

For each eventuality λ , all branches determine a unique tense according to table 4.A (there might be branches that do not determine any tense).

Proof. In the statement of the lemma we might have given reasons why these lists of κ_i , $\kappa_{i,j}$, and $\kappa_{i,j,k}$ are finite. Though this is clearly unnecessary. For note that it follows from DRT that every *DRS* is finite, in the sense that it has only finitely many conditions. And this clearly implies the finiteness of these lists.

In order to give a complete proof we should show that: (i) there is always a branch that determines a tense according to table 4.A, and (ii) all branches determining a tense determine the same tense. To prove (i) it is enough to see that for each combination of features *TP*, *TENSE*, and *STAT*, there is always a branch that fits the scheme given in table 4.A. This is illustrated in table 4.B. To prove (ii), we should take into account the following facts. The only way to introduce eventualities in a DRS is by means of the top node construction rule 2.A. Further, the only way to introduce conditions that relate eventualities is by means of the aforementioned rule, or by means of the rules for the reference time (all of them described in chapter 2). The third column (from left to right) in table 4.B shows all the possible conditions introduced by such rules. The rightmost column in that table shows the branches that can be constructed if the conditions in the third column (from left to right) are found in the DRS. It is not difficult to check that for each row that has more than one branch, either only one of them determines a tense, or they all determine the *same* tense. This completes the proof. \square

Tense	Feature Combination	List of Conditions $e' = \text{Ref. time}$	Accepted Branches
Simple past	$TP = -past$ $STAT = -STAT$ $TENSE = past$	$e' < e$ $t < n$ $e \subseteq t$	$e \subseteq t$ $t < n$
Future	$TP = -past$ $STAT = -STAT$ $TENSE = fut$	$e' < e$ $n < t$ $e \subseteq t$	$e \subseteq t$ $t < n$
Simple past	$TP = -past$ $STAT = +STAT$ $TENSE = past$	$e' \subseteq s$ $t < n$ $s \circ t$	$s \circ t$ $e' \subseteq s$ $t < n$ $e' < e$
Present	$TP = -past$ $STAT = +STAT$ $TENSE = pres$	$e' \subseteq s$ $t = n$ $t \subseteq s$	$t \subseteq s$ $e' \subseteq s$ $t = n$ $e' < e$
Future	$TP = -past$ $STAT = +STAT$ $TENSE = fut$	$e' \subseteq s$ $n < t$ $s \circ t$	$s \circ t$ $e' \subseteq s$ $n < t$ $e' < e$
Past Perfect	$TP = +past$ $STAT = -STAT$ $TENSE = past$	$e' < e$ $\mathbf{TPpt} < n$ $t < \mathbf{TPpt}$ $e \subseteq t$	$e \subseteq t$ $t < \mathbf{TPpt}$
Past Future	$TP = +past$ $STAT = -STAT$ $TENSE = fut$	$e' < e$ $\mathbf{TPpt} < n$ $\mathbf{TPpt} < t$ $e \subseteq t$	$e \subseteq t$ $t < \mathbf{TPpt}$
Past Perfect	$TP = +past$ $STAT = +STAT$ $TENSE = past$	$e' \subseteq s$ $\mathbf{TPpt} < n$ $t < \mathbf{TPpt}$ $s \circ t$	$s \circ t$ $e' \subseteq s$ $e' \subseteq s$ $t < \mathbf{TPpt}$ $e' < e''$ $t < e'$
Simple past	$TP = +past$ $STAT = +STAT$ $TENSE = pres$	$e' \subseteq s$ $\mathbf{TPpt} < n$ $t = \mathbf{TPpt}$ $t \subseteq s$	$t \subseteq s$ $e' \subseteq s$ $e' \subseteq s$ $t = \mathbf{TPpt}$ $e' < e''$ $t = e'$
Past Future	$TP = +past$ $STAT = +STAT$ $TENSE = fut$	$e' \subseteq s$ $\mathbf{TPpt} < n$ $\mathbf{TPpt} < t$ $s \circ t$	$s \circ t$ $e' \subseteq s$ $e' \subseteq s$ $\mathbf{TPpt} < t$ $e' < e''$ $e' < t$

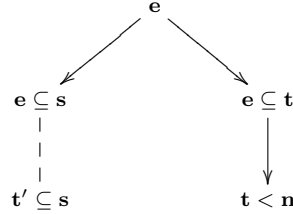
• e'' comes from a third sentence
 • The rightmost branch is due to the possibility of $e' = \mathbf{TPpt}$

Table 4.B: The Relation between the first and the second column is taken from table 2.2.5. The relation between the second and third column is given by the construction rule 2.A and the reference time rules. The rightmost column contains the possible branches of the respective eventuality that could be found in any arbitrary DRS.

Now, let us get done the translation of our example. Instruction (4.3) will predict the following modification in the \mathcal{II} :

\mathcal{II}
<i>Jean</i> [x] _{<i>f</i>}
$x = \mathbf{Jean}$
<i>the sweater</i> [y] _{<i>f</i>}
$y = \mathbf{the\ sweater}$
<i>the room</i> [z] _{<i>f</i>}
$z = \mathbf{the\ room}$
<i>take off</i> (t, x, y) comes from \mathbf{e} is $\langle -, -, +, + \rangle$
<i>warm</i> (t, z) comes from \mathbf{s} is $\langle -, -, -, + \rangle$
?HOLDSAT(<i>Jean</i> [x] _{<i>f</i>} , R_1) SUCCEEDS
? $x = \mathbf{Jean}$ SUCCEEDS
?HOLDSAT(<i>the sweater</i> [y] _{<i>f</i>} , R_2) SUCCEEDS
? $y = \mathbf{the\ sweater}$ SUCCEEDS
?HOLDSAT(<i>the room</i> [z] _{<i>f</i>} , R_3) SUCCEEDS
? $z = \mathbf{the\ room}$ SUCCEEDS

Let us do in some detail the step carried out by instructions (4.4). We begin with line ‘*take off*(t, x, y) comes from \mathbf{e} is $\langle -, -, +, + \rangle$ ’. In looking up in DRS (3.1) the conditions related with \mathbf{e} by \subseteq or \circ , we find $\mathbf{e} \subseteq \mathbf{t}$ and $\mathbf{e} \subseteq \mathbf{s}$. Performing the search of the conditions related with \mathbf{t} and \mathbf{s} by $<$ or $=$, we find $\mathbf{t} < \mathbf{n}$. This, according to table 4.A, implies that \mathbf{e} is in the past tense. It is worth taking a closer look at the list just made, that I present here in a tree-like diagram:



Note that the left horn does not lead to a valid list according to the definitions in lemma 1. This is clear because the list $\kappa_{i,1}, \dots, \kappa_{i,r_i}$ **should** consists of the eventualities related to κ_i by either $<$ or $=$. In the left horn, however, the relation between \mathbf{s} and \mathbf{t}' is that of \subseteq , which does not lead to the inclusion of the condition $\mathbf{t}' \subseteq \mathbf{s}$ in the list. Therefore, the left horn does not lead to a branch. Thus, there is only one branch, and this is $\mathbf{e} \subseteq \mathbf{t}$. This, according to table 4.A, implies that \mathbf{e} is in the past tense.

Now, in working with line ‘*warm*(t, z) comes from \mathbf{s} is $\langle -, -, -, + \rangle$ ’, we find the branches $\mathbf{t}' \subseteq \mathbf{s}$, and $\mathbf{e} \subseteq \mathbf{s}$. Both branches determine that \mathbf{s} is in the past tense.

From these results and the definitions in [19, ch. 8], and the rules we introduced in §3.6, we find the following IC’s, which we introduce in \mathcal{II} :

?HOLDSAT(*on*[x, y]_{*f*}, R_4), HAPPENS(*take off*[x, y], R_4), $R_4 < \mathbf{now}$ SUCCEEDS
 ?HOLDSAT(*warm*[z]_{*f*}, R_5), $R_5 < \mathbf{now}$ SUCCEEDS

Now, it remains to link the times R_i on which we require the fluents representing objects to hold with the times R_j on which we require the eventualities to hold. We can do it by taking certain information from the IC's in \mathcal{II} and adding it to \mathcal{II} . We do this by applying the following instructions to each IC:

If line is	Follow instructions
(4.5) $? \text{HOLDSAT}(f, R)$ SUCCEEDS	(i) Decode the variables in f . (ii) For each variable x decoded in the previous step introduce line ' $x \Rightarrow R$ ' in \mathcal{II} .

Then, after all the IC's have been scanned, we perform the following operation in \mathcal{II} :

$$(4.6) \text{ If } x \Rightarrow R \text{ and } x \Rightarrow R' \text{ are both in } \mathcal{II}, \text{ introduce in } \mathcal{II}: \\ ?R = R' \text{ SUCCEEDS.}$$

This procedure yields the following \mathcal{II} for our example:

\mathcal{II}
<i>Jean</i> [x] _{f}
$x = \mathbf{Jean}$
<i>the sweater</i> [y] _{f}
$y = \mathbf{the sweater}$
<i>the room</i> [z] _{f}
$z = \mathbf{the room}$
<i>take of f</i> (t, x, y) comes from \mathbf{e} is $\langle -, -, +, + \rangle$
<i>warm</i> (t, z) comes from \mathbf{s} is $\langle -, -, -, + \rangle$
?HOLDSAT(<i>Jean</i> [x] _{f} , R_1) SUCCEEDS
? $x = \mathbf{Jean}$ SUCCEEDS
?HOLDSAT(<i>the sweater</i> [y] _{f} , R_2) SUCCEEDS
? $y = \mathbf{the sweater}$ SUCCEEDS
?HOLDSAT(<i>the room</i> [z] _{f} , R_3) SUCCEEDS
? $z = \mathbf{the room}$ SUCCEEDS
?HOLDSAT(<i>on</i> [x, y] _{f} , R_4), HAPPENS(<i>take of f</i> [x, y], R_4), $R_4 < \mathbf{now}$ SUCCEEDS
?HOLDSAT(<i>warm</i> [z] _{f} , R_5), $R_5 < \mathbf{now}$ SUCCEEDS
$x \Rightarrow R_1$
$y \Rightarrow R_2$
$z \Rightarrow R_3$
$x \Rightarrow R_4$
$y \Rightarrow R_4$
$z \Rightarrow R_5$
? $R_1 = R_4$ SUCCEEDS
? $R_2 = R_4$ SUCCEEDS
? $R_3 = R_5$ SUCCEEDS

To finish up the construction of the IC, we need to go over the lines of \mathcal{II} , collect all the lines beginning with ?, and build up an IC with all of them. The

application of this step in our example delivers the following (huge) IC:

?HOLDSAT(*Jean*[x] _{f} , R_1),
 $x = \mathbf{Jean}$,
HOLDSAT(*the sweater*[y] _{f} , R_2),
 $y = \mathbf{the\ sweater}$,
HOLDSAT(*the room*[z] _{f} , R_3),
 $z = \mathbf{the\ room}$,
HOLDSAT(*on*[x, y] _{f} , R_4), HAPPENS(*take off*[x, y], R_4), $R_4 < now$,
HOLDSAT(*warm*[z] _{f} , R_5), $R_5 < now$,
 $R_1 = R_4$,
 $R_2 = R_4$,
 $R_3 = R_5$ SUCCEEDS

This delivers the three important steps of the semantic representation in EC.

It is not difficult to see that the semantic representation just produced by our translation algorithm is equivalent to the representation given in the previous chapter.

4.2 Test on Other Past Tense Sentences

Consider the sentence

(4.7) A delegate arrived. She registered.

represented by the following DRS:

e n t x e' t' y
e \subseteq t
t $<$ n
<i>delegate</i> (x)
e : <i>arrive</i>(x)
e' \subseteq t'
t' $<$ n
e $<$ e'
e' : <i>register</i>(y)
$x = y$

In order to create the language in EC for this DRS we follow instructions (4.1), obtaining the following \mathcal{IT} :

\mathcal{II}
$delegate[x]_f$ $x = \mathbf{delegate}$ $arrive(t, x)$ comes from \mathbf{e} $register(t, y)$ comes from \mathbf{e}'

Here we anticipate a trouble. In the DRS, the condition $x = y$ will make sure that an appropriate referent is assigned to y . But we have not yet specified instructions to take this information from the DRS, and if we do not do so, y will be a free variable —i.e. the representation in EC will be wrong. We need to add the following line to instructions (4.1):

If DRS Condition is: Follow instruction:

$?x = y$ SUCCEEDS Introduce in \mathcal{II} :
 $x = y$.

We assume we have introduced $x = y$ in \mathcal{II} .

Assume the following database:

$\mathbf{e} \mapsto \langle -, -, +, + \rangle$
 $\mathbf{e}' \mapsto \langle -, -, +, + \rangle$

We then follow instructions (4.2), obtaining the following scenario:

$\{\text{HOLDSAT}(C_1[x]_f, t) \rightarrow \text{Inititates}(arrive[x]_e, t, \overline{C}_1[x]_f),$
 $\text{HOLDSAT}(C_2[y]_f, t) \rightarrow \text{Inititates}(register[y]_e, t, \overline{C}_2[y]_f)\}$

We can continue with instructions (4.3). Applying these instructions we obtain the following \mathcal{II} :

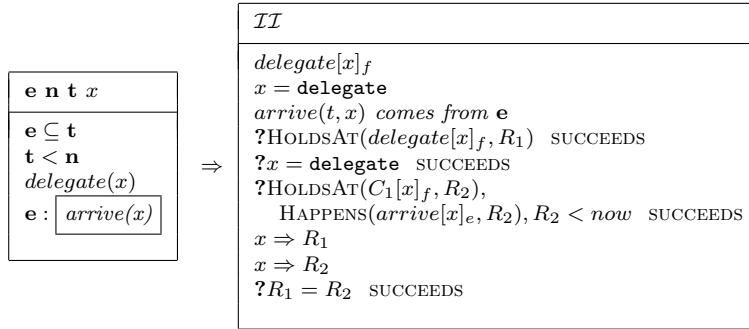
\mathcal{II}
$delegate[x]_f$ $x = \mathbf{delegate}$ $arrive(t, x)$ comes from \mathbf{e} $register(t, y)$ comes from \mathbf{e}' $? \text{HOLDSAT}(delegate[x]_f, R_1)$ SUCCEEDS $?x = \mathbf{delegate}$ SUCCEEDS $?x = y$ SUCCEEDS

We need now to follow instructions (4.4). With those instructions we find the branch $\mathbf{e} \subseteq \mathbf{t}$ related with \mathbf{e} . Thus, according to table 4.A, this eventuality is in the past tense. We do something similar with \mathbf{e}' . The branch determined by it is $\mathbf{e}' \subseteq \mathbf{t}'$, which implies that \mathbf{e}' is also in the simple past tense. Carrying out instructions (4.4), (4.5), and (4.6), we obtain the following \mathcal{II} :

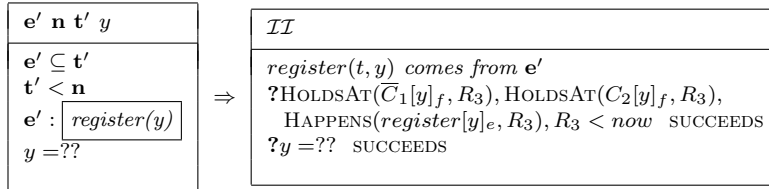
\mathcal{II}
$delegate[x]_f$ $x = \mathbf{delegate}$ $arrive(t, x)$ comes from \mathbf{e} $register(t, y)$ comes from \mathbf{e}' $?HOLDSAT(delegate[x]_f, R_1)$ SUCCEEDS $?x = \mathbf{delegate}$ SUCCEEDS $?x = y$ SUCCEEDS $?HOLDSAT(C_1[x]_f, R_2), HAPPENS(arrive[x]_e, R_2), R_2 < now$ SUCCEEDS $?HOLDSAT(\overline{C}_1[y]_f, R_3), HOLDSAT(C_2[y]_f, R_3),$ $HAPPENS(register[y]_e, R_3), R_3 < now$ SUCCEEDS $x \Rightarrow R_1$ $x \Rightarrow R_2$ $?R_1 = R_2$ SUCCEEDS

It is straightforward to get the IC from this \mathcal{II} . An important comment is in place. Note that, according to instructions in §3.6, the IC-line in \mathcal{II} for the second sentence includes a $HOLDSAT$ clause for the result state of the previous sentence. At first sight this might strike as unnecessary, since the final IC would make that clause true in any case. However, it turns out to be of paramount importance for the temporal anaphora. It can be formally proven, by means of EC axioms, that the introduction of this clause implies the constraint $R_2 < R_3$ to hold. Doing so, however, would take us too long. I will just give some brief remarks based on reasoning in a minimal model that would guide such formal proof. Firstly, note that the IC require both $HOLDSAT(C_1[x]_f, R_2)$ and $HOLDSAT(\overline{C}_1[x]_f, R_3)$ to hold. This, clearly, implies that $R_2 \neq R_3$. We are left with two options: either $R_2 < R_3$ or $R_3 < R_2$. Note that there is no event e such that a clause like either $INITIATES(e, t, C_1[x]_f)$ or $TERMINATES(e, t, \overline{C}_1[x]_f)$ belongs to the scenario. Thus, since we require $HOLDSAT(C_1[x]_f, R_2)$ to hold, we need to update the scenario with a clause $INITIALLY(C_1[x]_f)$. Since there is no event in the scenario whose occurrence clip the clause $HOLDSAT(C_1[x]_f, t)$ with $0 \leq t < R_2$, then by **Axiom 1.1** and **Axiom 2.1** it follows that $HOLDSAT(C_1[x]_f, t)$ is true, for $0 \leq t < R_2$. This clearly implies that $HOLDSAT(\overline{C}_1[x]_f, R_3)$ cannot be true if $R_3 < R_2$. However, since the IC requires it to be true, then we need to have $R_2 < R_3$. So much for the temporal anaphora.

This example is also analyzed in [7]. We can elaborate it along similar lines. That is, we can show that carrying out the translation of example (4.7) sentence by sentence —i.e., getting the DRS for the first sentence and translating it in EC, and then getting the DRS for the second sentence and translating it to EC— achieves the same result as translating it altogether as we did before. We have for the first sentence:



Plus the scenario and IC determined by it. For the second sentence we have:



Plus the scenario and IC determined by it. The only difficult part in the previous representation is the anaphoric pronoun y . Since the second sentence is to be interpreted against the context given by the first, we can use a number of procedures to give an appropriate referent to y . These procedures are explained in [9, §1.1.1, p. 66 and ff]. Nothing precludes us to use these same procedures in \mathcal{II} . So, we can assume that $?y = ??$ SUCCEEDS is resolved with $?y = x$ SUCCEEDS. It is not difficult to see that updating the first scenario with the first IC, and then enhancing these result scenario with the second scenario and updating it with the second IC produces the same result scenario as before.

4.3 Comments on DRT and EC

The translation algorithm here designed was successfully applied to two mini-discourses:

(4.8) Jean took off the sweater. The room was warm.

(4.9) A delegate arrived. She registered.

The former is a sequence of an event (an achievement) followed by a state. The latter is a sequence of two events (two achievements). We still need to test the algorithm in sequences of sentences of the form ‘state + state’ and ‘state + event’. (There is a minor difficulty with a sequence of the form ‘state + event’ that should be mentioned. DRT does not handle this sort of sequences—at least not the version of DRT in [9]. In this case, though, it is not difficult to improve on DRT’s definitions in order to handle it.) Also, all sort of ‘correct’ sequences of aktionsarten should be tested, and the algorithm should be extended to the

progressive, the past perfect tense, etc. We also need to test it with more than two sentences.

It will also be important to assess whether DRT can be enhanced to deal with aktionsarten in a more natural way, and from here, we could seek to improve on the database for the aktionsarten (Cf. §4.1.2). This might be one point where we could start looking for something better for the translation algorithm. I mean the following. The aktionsarten of a sentence does not seem to come from the verb, but from the whole VP, and this process does not seem to be compositional (Cf. §6.1.2). It stands in need of exploration how much of this non-compositionality can be handled by DRT, but it might as well be an argument to seek for a different approach where this non-compositionality squares better.

The limits of the path we have followed are precisely the limits of DRT. One mayor difficulty in the path is the future tense. DRT is not well suited to handle with this phenomenon.¹ Examples like:

(4.10) John flies to Chicago tomorrow,

(4.11) Bill will throw himself off the cliff,

(4.12) Bill is going to throw himself off the cliff.

do not seem to be easily handled by DRT. The first one is a future tense, but its grammatical form is present. Thus, in the syntactic tree of (4.10), the top node will contain the information of the present tense. The information of the future tense only comes from the adverb ‘tomorrow’. If we were to represent this sentence in DRT, we will represent first the top node and then the adverb. The relation between reference time, temporal perspective point, and eventuality would be fixed by the algorithm according to the present tense. This is not the way we want to represent (4.10). Other problem with DRT is that the distinction between (4.11) and (4.12) resides on the satisfaction of a plan. But DRT cannot represent a plan, and thus it does not seem clear whether DRT can cope with the appropriate meaning of these cases.

Cases like (4.10-4.12) are of great interest to EC. So, if we want to analyze them in a formal way, we need to come up with a better algorithm than the one proposed here. Algorithm that does not rely on DRT, since this latter is unable to do justice to them. This might serve as a motivation to explore different alternatives to give EC a translation from natural language temporal discourse, other than via DRT.

¹I rely for this claim on a personal communication with Fritz Hamm.

Part II

Construction Grammar and Event Calculus

Chapter 5

Cognitive-Functional Linguistics

5.1 The New Psychology of Language

Since Generative Grammar, the notion of language has been given a precise answer. A language is a set of sentences generated by a finite set of rules from a finite lexicon. This set of rules is said to be codified in the mind/brain, in the form of a ‘mental organ’ independent of any other cognitive function. This last tenet has been given the name of *Autonomy of Syntax*.

Recently, some tendencies in psychology and psycholinguistics have argued conjointly against this tenet. One of the main arguments raised against it is that it was introduced just for the sake of ‘mathematical elegance’, and this reason is not compelling enough to make it into a leading principle in the psychological study of language (Cf. [17, p. x]). It is also argued that the data customarily used in this approach is restricted to disembodied sentences and written discourse. This is inappropriate, according to this new tendencies, for it leaves out what is considered to be central to the phenomenon of language, that is, real discourse situations —discourse that has its own particular characteristics.

Autonomy of syntax is precisely the negation of what these new tendencies consider to be the correct path to understanding language. Namely, natural language is to be understood by means of both its function and more general biological dispositions (Cf. [17], p. xi). Within this new approach, cognitive and functional considerations are allowed to be brought to bear on syntax. The notion of language that emerges from here deserves some attention. Note from the outset that a language can still be considered as a set of sentences generated by a finite set of rules. However, the set of atoms the rules are applied to, and the characteristics of the rules, are given different interpretations. That is, both atoms and rules are understood in terms of its function to convey meaning. The concept often used to describe what a language is is that of ‘symbol’. The following quote from [11, p. 2], is quite insightful as to what grammar is within

the approach in question:

[...] grammar is per se a *symbolic* phenomenon, consisting of patterns for imposing and symbolizing particular schemes of conceptual structuring. It is held that lexicon, morphology, and syntax form a continuum fully describable as assemblies of *symbolic structures* (form-meaning pairings), and consequently, that all valid grammatical constructs have conceptual import.

There is a marked difference in emphasis regarding the way data is examined in both approaches. The central question we ask ourselves when presented with a sentence is not whether it sounds odd or not. But rather we ask ourselves what the meaning of that sentence is, and which conceptual structures license that sentence having that meaning. Thus, it is not surprising that the set of data this new approach uses contains idioms, metaphors, and irregularities, and that they form many of the cases of analysis.

The difference in emphasis and the kind of enterprise undertaken is expressed by Tomasello [17, p.xii] in the following quote:

The ultimate goal is not to create a mathematically coherent grammar that normatively parses the linguistic universe into grammatical and ungrammatical sentences, but rather to detail the “structured inventory of symbolic units” that make up particular natural languages.

This ‘new psychology of language’ is the approach under which we will work in this second part of the thesis. The main goal here is to use formal tools in the explanation of grammar as considered in the aforementioned approach. It is important to note that this is an exploratory step, as there are —to my knowledge— no other previous attempts to formalize this particular approach to language.

Be that as it may, the change in the notion of language, and the new added emphasis on the influence of cognitive structures on the form of language, all bring about changes on the way formal methods are applied to the study of natural language meaning. Several differences need to be assessed. The most important —at least for the present work— are the characteristics of the atoms the language is built up from, and the connection between meaning and form. In the rest of this chapter I will devote my attention to these issues in turn, in order to bring out the consequences of the change in approach, and to ‘reschedule’ the framework we used in the first part of this thesis to attach meanings to sentences.

5.2 The New ‘Atoms’: Constructions

Recall that in the approach we are considering in this part the analysis resides on a description of the symbolic units of the language. This does not bring about a reduction on the number of elements that are already recognized as constituents of the language. (These traditional types of symbolic elements are words, markers on words¹, word order, and intonation.²) On the contrary, other

¹Markers on words are, for instance, the particle *-s* for making plurals, that exist in some languages like English, Spanish and French.

²Cf. [17, p. xvi].

symbolic units need to be recognized if we want to describe the whole inventory of symbolic tools a language possess, and the whole range of linguistic data.

Those new elements in the inventory of symbolic units are called *constructions*, and are the distinctive mark of Construction Grammar (CG from now on). Here is how Adele E. Goldberg defines a construction:

A construction is defined to be a pairing of form with meaning/use such that some aspect of the form or some aspect of the meaning/use is not strictly predictable from the component parts or from other constructions already established to exist in the language.³

The defendants of CG spend a considerable amount of time arguing in favor of the existence of constructions different from morphemes. Some of these arguments are summarized in §6.1, but I will not do justice to a good defence of CG—that is not the main objective of this thesis.

In this work I will focus on a few number of constructions: argument structure constructions. Moreover, I will only consider two of these constructions: the Ditransitive Construction, and the Caused Motion Construction. To illustrate them, consider the following examples:

(5.1) Margaret baked Peter some cookies.

(5.2) Martin sneezed the napkin off the nightstand.

Example (5.1) used the verb ‘bake’ with three arguments. Normally, this verb has only two. This is what makes this sentence particular. However, we can attach a clear meaning to it: Margaret baked some cookies with the intention to give them to Peter. Note that thinking of this sentence in this particular way is what helps us make sense of the recipient role in it, which is not provided by the verb ‘bake’. This way of thinking of this sentence is precisely the sense of the ditransitive construction. This construction would have been illustrated clearer by means of example (5.3), but this last example does not have the ‘controversial’ flavor of (5.1):

(5.3) John gave Mary a present.

As for example (5.2), we are using here the verb ‘sneeze’, which is an intransitive verb. But we can make sense of (5.2) in the way that Martin’s sneezing caused—due to the strong exhalation of air—the napkin to go off the nightstand. This way of understanding (5.2) is the Caused Motion Construction, and it is the one that provides the sentence with two argument roles that the verb ‘sneeze’ does not provide. This construction could have been illustrated by:

(5.4) Sam sent Cynthia to the market.

In the next chapter, I will present a brief overview of the traditional arguments in favor of argument structure constructions.

³[6, p. 68].

5.3 Syntax and Semantics

A few words concerning the relationship between syntax and semantics are in place here. Four different positions regarding such relationship can be distinguished:

1. A first position can be identified as the complete precedence of syntax over semantics. This position emerges from the Chomskyeen approach to language, and mainly from its Autonomy of Syntax tenet. This tenet says that syntax is autonomous from semantics, in the sense that syntactic representations are considered in isolation, without any intervention of semantics, or any other cognitive function. Once the syntactic representations are in place, semantics can take over and provide semantic representations built upon syntactic representations. This position is clearly illustrated by the semantics of formal languages. There, the syntax is completely specified by means of recursive rules, and is made before any semantic definition is given. Once syntax is in place, a semantic rule is attached to each syntactic rule, providing thus a semantic representation to all well-formed formulas of the language. Another example is Lewis' system, which explicitly relies on syntax [12, p. 169]:

‘On the hypothesis that all natural or artificial languages of interest to us can be given a transformational grammars of a certain not-very-special sort, it becomes possible to give very simple general answers to the questions:

- (1) What sort of thing is a meaning?
- (2) What is the form of the semantic rules whereby meanings of compounds are built up from the meanings of their constituent parts?

2. A second position is more modest with respect to the autonomy and precedence of syntax. Syntax still precedes semantics, although syntax is open to some modifications required by the semantic theory. A clear case is DRT, in which we require the top node of the phrase marker to have a lot of semantic information, and where the phrase markers are progressively modified into DRS's.
3. A third position regards syntax as an instrument of semantics. Thus, the precedence is inverted. Syntax is understood as ‘patterns for imposing and symbolizing particular schemes of conceptual structuring’. So, it is in function of conceptual structuring that we understand syntax. Semantics is independent from syntax, and is given beforehand. This is the position presented in §5.1. As I said, the order of precedence is inverted, but there is still place for some independence of syntax, as explained by Tomasello [17, p. xii]:

This functional approach does not mean that all structures in language are determined by function in the sense that they are iconically related to their meanings, as many generative grammarians misconstrue the claim [...] The claim is simply that both cultural artifacts and biological structures are understood primarily in terms of their functions, and so to leave them out is to miss their point entirely.

4. The last position is the one upholding a radical supremacy of semantics. For this position, syntax is just an epiphenomenon of semantics. Cognitive models can be given in isolation —that is, completely independent from language. Besides, there is no well or ill formed sentences. Everything goes as long as we are able to give it a semantic representation —in the form of a cognitive model.

In the first part of the thesis we were working under the second position, and we are moving on to the third position. There is no space here to assess the changes and subtleties brought about by the inversion in precedence between syntax and semantics. The foregoing comments can be taken just as a heads up that something changed, and that we need to assess such a change. Furthermore, this part of the thesis assumes the precedence of semantics over syntax. It is not easy to say whether we will stay in the third or the fourth position described above. The leading idea here is that EC can take the lead in the formalization of language, and in particular, of CG. How much of CG can be formalized with EC is not known yet, but I want to close this chapter with a brief argument that points out that EC might not be enough, and that maybe we also should stay in the more modest third position. The argument has to do with the syntactic representation of argument structure constructions. Consider the ditransitive construction. On the one hand, Goldberg [5] regards it as completely determined by the sentence type *NP V NP NP*. It is worth noting that it is this approach that we consider in here. On the other hand, Michaelis [13] argues that this might not be that simple:

As Fillmore and Kay (1993: Chapter 8) point out, the constraints that define the ditransitive construction are operative whether the recipient argument is realized as a postverbal accusative NP, as in an active sentence, or as a preverbal nominative NP, as in a passive sentence, e.g., *We were given the book*. For this reason, we will assume, as they do, that the ditransitive is a schematic (i.e., non-lexically specified) verb entry, in which the grammatical functions of the agent and recipient arguments are unspecified.

I quoted this passage because I do not understand altogether the reasons for this change. But if such a change is necessary, or even more convenient, we will face problems that are *prima facie* unsurmountable by EC. For instance, consider examples (5.5-5.6):

(5.5) John gave Mary the book.

(5.6) Mary was given the book by John.

According to Michaelis, the argument structure of these examples is the ditransitive construction. Clearly, the order of the NP's is not enough to link 'Mary' with the recipient role. Which semantic information bounded to causal relations is able to determine that Mary is the recipient role in both sentences? Why should there be any principled difference between the semantics of 'John' and 'Mary'? If there were some difference, could it work out the same way if we interchange in (5.5-5.6) the roles of 'John' and 'Mary'? It seems that 'John'

and ‘Mary’ are indistinguishable in semantic terms. This means that no semantic information, that is not based on the structure of the sentence, is sufficient to determine the recipient role of the sentences in question. If this intuition is correct, then in order to formalize CG we would need more than EC, and even more than any purely semantic information.

Chapter 6

Bird's eye view on Construction Grammar

6.1 Argument Structure Constructions

I will briefly review in this section the most important points in the argumentation given by Adele Goldberg [5] in favor of a CG approach. The leading question of Goldberg is this: ‘What is the nature of verb meaning and what is its relation to sentential meaning?’ This question has been given an answer under the Fregean tradition: the relation between sentential and verb meaning can be made explicit by the principle of compositionality. Accordingly, the meaning of a verb is given by its contribution to the meaning of the sentences in which it appears. Frege identified the meaning of a verb with a n -ary predicate that, along with n arguments, yielded a proposition —which is the meaning of the sentence in which the verb appears. Goldberg challenges this picture of verb meaning that emerges from the principle of compositionality. She posits that the verb's argument structure is a linguistic entity on its own right and that it has meaning independent of the meaning of the verb and the meaning of each of the arguments. In order to make this more concrete, we can work with an example. Consider the following sentence:

(6.1) John sneezed the napkin off the table.

The meaning of (6.1) depicts a situation in which John causes the napkin to move by sneezing.¹ The form of the argument structure of this sentence is ‘X causes Y to move Z’.² The controversial claim is that this argument structure is independent of the verb, and has meaning on its own. Since the meaning of this ‘creature’ is not based on word meaning —because we got into it by stripping

¹This is an odd sentence, since ‘sneeze’ is an intransitive verb. The interesting thing is that as long as we can make sense of this sentence, we should account for the fact that we can do so.

²Where X is John, Y is John's sneeze and Z is the napkin.

the words away— it is hard to fit it in a compositional picture. The meaning of the sentence is no longer only predictable from the meaning of the words that appear on it and the way they are put together, but we need in addition the meaning of the (independent) argument structure. Moreover, particular constraints or licenses are ascribed to the argument structure instead of the verb, and thus argument structure plays an important role in grammar. More on this below.

We can take a look at the argument structures Goldberg is mainly interested in:

Ditransitive: X causes Y to receive Z,

Caused Motion: X causes Y to move Z,

Resultative: X causes Y to become Z,

Intransitive Motion: X moves Y,

Conative: X directs action at Y.

These entities are called *constructions*, and are intended to fit in the general definition of constructions given in Construction Grammar.

6.1.1 Construction Grammar

Some general comments about constructions are in place. To begin with, there are constructions whether one likes it or not: it is not difficult to see that morphemes satisfy the aforementioned definition of a construction. Further, CG is generative, since it tries to account for the whole infinite range of expressions that are allowed and the infinite range of expressions that are disallowed. However, unlike traditional grammars, CG does not posit a privileged class of ‘core cases’. All expressions of a language stand in the same level. For instance, example (6.1) is an odd construction, but is not treated as an exception. In order to handle with these cases, non morphemic constructions are posited, thus aiming to account for the whole range of empirical data. Another remark is that CG does not make a strict division between lexicon and syntax. They are regarded as the same type of declarative represented data structure. Finally, CG is not transformational. A deep structure of (6.1) could have been posited: something like ‘John’s sneeze caused the napkin to get off the table’. This way out is rejected by CG. There is no sentence’s deep structure.

6.1.2 A comparison with lexicosemantic rules

The main insight for the comparison between CG and lexicosemantic rules (LSR) is that changes in argument structure are crucially semantic. In a LSR approach, the argument structure is taken to be uniquely predictable from the semantic representation of the verb. Therefore, if a verb is correctly used with different argument structures, this reflects differences in the semantic representation of

the verb. On the contrary, in the approach proposed by Goldberg, argument structure is (relatively) independent from the verb. These two approaches are compared with respect to a series of points with the aim to compel the reader that the CG approach is more plausible than the LSR approach. These points are taken in turn in what follows.

Implausible verb senses are avoided

Consider again example (6.1) and compare it with ‘John sneezes’. The latter is an intransitive construction—in fact, ‘sneeze’ is an intransitive verb. The former is a ditransitive construction. Since we have two different argument structures, a LSR approach predicts different semantic representations of ‘sneeze’, one for the transitive, ‘sneeze₁’, and another for the ditransitive, ‘sneeze₂’. This distinction would make sense if there were a language with two different words, one with the meaning of ‘sneeze₁’, and another with the meaning of ‘sneeze₂’. However, this seems very implausible.

Under a CG approach, differences in meaning are due to different constructions. So, in both sentences, ‘sneeze’ has the same meaning, and the differences in meaning of the whole sentence can rather be attributed to the different constructions.

Circularity is avoided

Goldberg points out a circularity embedded in the LSR approach. To begin with, she presents eight sentences with the verb ‘kick’, all of them with different argument structure.³ Some of them are:

(6.2) Patrick kicked the ball,

(6.3) Patrick kicked his foot against the chair,

(6.4) Patrick kicked Mary a ball.

She claims that both the assertion that ‘kick’ has a particular n -argument meaning and the explanation that ‘kick’ has n arguments come from the fact that ‘kick’ occurs in that particular n -argument structure. To make it clearer, the circularity is that ‘kick’ is claimed to have a n -argument meaning because it occurs with n arguments, and at the same time it is claimed that ‘kick’ occurs with n arguments because it has a n -argument meaning.

Since in the CG approach the arguments are associated directly to the construction, the circularity is avoided. For CG does not claim that ‘kick’ has a n -argument meaning because it occurs with n arguments.

Explaining non-compositional cases

The principle of compositionality is criticized by Goldberg by means of two phenomena: the Dutch impersonal passive construction and the way construction in English. I will only discuss the Dutch impersonal passive construction.

³[5], p. 11.

The Dutch impersonal passive construction: Consider the following examples:

- (6.5) (a) Er werd gelopen,
 There was run,
 (b) *?Er werd naar huis gelopen,
 There was run home,
 (c) Er werd voortdurend naar huis gelopen,
 There was constantly run home.

The constraint seems to be that the impersonal passive is allowed for atelic situations (like those in (6.5a, c)), and not for telic situations (like the one in (6.5b)). This implies that this is a constraint on the whole expression, and since the verb ('gelopen') is the same in all cases, it follows that the constraint cannot be expressed as a lexical constraint. Postulating a telic and an atelic meaning of the verb 'gelopen' would allow us to keep this constraint as a lexical constraint. However, such a move would not allow us to explain the behavior of the auxiliaries 'zijn' and 'hebben' (the former used with telic and the latter with atelic verbs). They depend exclusively on the aspect of the verb, independent of the aspect of the whole sentence. Thus, a telic sense of (6.5a) is disallowed, which makes less plausible the possibility of postulating different meanings of the verb 'gelopen'.

Goldberg claims correctly that the principle of compositionality implies that the meaning of an expression is determined only by the meaning of the lexical units that occur in it. There is no other kind of meaning that contributes to the meaning of the expression. (If we said that the meaning of an expression is determined also by the meaning of a subexpression we would not add anything, since the meaning of that subexpression is in turn determined by the meaning of the lexical elements that occur in it.) The composition rules are, in this sense, empty with respect to its contribution to the meaning of the expression—they just sort of *combine* meanings, but they do not *add* meaning.

In trying to make sense as to how the Dutch impersonal passive construction bears upon compositionality, we can think that the aspect of a situation is an integral part of the meaning of a sentence. As an element of meaning, compositionality implies that aspect should be determined at the lexical level—and there is no other option that ascribing it but to the verb.⁴ Here is where the problem lies, since aspect seems to be required distinctly for the verb and for the whole sentence. Examples (6.5a-c) motivate a change in the aspect of the whole sentence, whereas the auxiliary selection phenomenon motivates a fixed verb aspect. Thus, where does the aspect of the whole sentence come from?

Under the CG approach the answer is: the aspect of the whole sentence comes from the aspect of the construction. Goldberg has the following alternative claim

⁴This chain of reasoning works out as long as we consider the aspect of a sentence not to be a composition of other sort of meanings. It is true, though, that it sounds strange to say that aspect is made up from other things that are not aspectual meanings.

to the principle of compositionality —which she claims to be a weak form of compositionality: ‘The meaning of an expression is the result of integrating the meanings of the lexical items into the meanings of constructions’.⁵

Supportive evidence from sentence processing

Goldberg upholds the hypothesis that the use of the same meaning of a verb in different argument structures do not show the same processing effects that cases of real lexical ambiguity do. Therefore, it seems plausible that we can measure the LSR thesis that a difference in argument structure implies a difference in verb’s meaning. For, in the face of a couple of sentences with different argument structure like:

(6.6) Bill loaded the truck onto the ship,

(6.7) Bill loaded the truck with bricks,

the phenomenon predicted by LSR is that of lexical ambiguity, and not that of same core meaning occurring in different argument structures. However, Carlson and Tanenhaus found that the processing times of sentences like (6.6-6.7) are different from sentences with real semantic ambiguity, like:

(6.8) Bill set the alarm clock onto the shelf,

(6.9) Bill set the alarm clock for six.

Their hypothesis is this: If a reader or hearer initially selects an inappropriate sense of an ambiguous word like ‘set’, the processing load will increase. On the other hand, if an inappropriate constructional use is selected, the reanalysis will be relatively cost free since the sense of the verb remains constant and the verb’s participant roles are already activated. The empirical evidence provided by Carlson and Tanenhaus, summarized in [5, p.18] seems to support this thesis.

6.2 Verbs and Constructions

6.2.1 Verb meaning

Verb’s meaning is defined with respect to a *frame* or background. A frame embodies a sort of world and cultural knowledge, and is highly structured. Let me explain the notion of a frame, and one of the important concepts associated with it —namely, the concept of *profiling*— by means of an example of a noun. Consider the term ‘hypotenuse’. This term is an Euclidean geometric concept, so it is possible to draw a mental picture of it. The interesting question regarding this mental picture is this: Can you imagine what a hypotenuse is without imagining the whole right triangle? It seems that the answer is: no. The triangle (and the flat plane it is included in) is a *frame*, and the terms ‘hypotenuse’ and

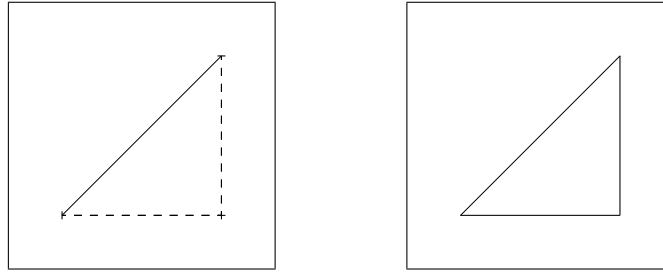


Figure 6.2.1: The frame semantics of the terms ‘Hypotenuse’ (left) and ‘Right Triangle’ (right). Adapted from [5], p. 26.

‘right triangle’ share such a frame, but *profile* different parts of that frame. See figure 6.2.1.

With this in mind, we can turn to our discussion of verb’s meaning. To begin with, we should note that verb’s meaning and noun’s meaning should not be too different if we are to explain the phenomenon of nominalization. This is why it is not altogether strange to say that verb’s meaning is also based in frame semantics. Moreover, the fact that verb’s meaning depends on world and cultural knowledge should easily come from examples like *renege*, *marry*, *boycott*, *riot*, *languish*, *lamine*, etc. Thus, in Goldberg’s account, the meaning of a verb is defined with respect to a frame, and the profiling of certain aspects of it. See §6.2.3.

6.2.2 Construction meaning

The construction meaning is a particular combination of roles which designate humanly relevant scenes. Following Langacker, Goldberg argues that construction’s meanings are certain recurrent and sharply differentiated aspects of our experience which we normally use to structure our conceptions insofar as possible. Construction’s meaning is the same—or ‘remarkably similar’—to some ‘general purpose verbs’. So claims about the latter imply claims about the former. For instance, verbs like ‘do’, ‘go’, ‘make’, ‘put’ are not only learned early crosslinguistically, but are also the most commonly used verbs in children’s speech. This claim shows that these verbs have a ‘basic’ status. Goldberg uses this fact to make the point that construction’s meanings—via their similarity to these verbs’ meaning—designate scenes which are semantically privileged in being basic to human experience.

Construction meaning is further shown to be polysemic. Constructions are typically associated with a family of closely related senses rather than a single one. Goldberg presents an example of construction meaning polysemy: the

⁵[5], p. 16.

ditransitive construction. She says that this construction imply that the agent argument acts to cause transfer of an object to a recipient. But it does not always imply that the transfer is successful. There is a range of expressions in which there is a transfer component, but it is either in the form of an intention, or something occurring in the future, or something precluded, etc. The idea is that there is a constellation of related senses of this ditransitive construction, that turns around the basic meaning of a transference.

Goldberg rejects the approach that we could stick with the construction's central sense as the only construction's meaning, and leave the related variations as part of the verb's meaning. The reason for the rejection is that it is highly unlikely that we can get a central meaning that is at the core of all the possible senses of the ditransitive construction.

6.2.3 Participant roles

The participant roles of a verb are defined as those in the frame semantics associated with the verb. The participant roles are lexically determined and highly conventionalized, they cannot be altered by context. Some of these participant roles are *profiled*, in the sense that they are obligatory accessed and function as focal points within the scene, achieving a special degree of prominence. The example presented is that of 'rob' vs. 'steal'. Both have three argument roles: 'thief', 'target', and 'goods'. However, 'rob' profiles only 'thief' and 'target', whereas 'steal' profiles only 'thief' and 'goods'.

6.2.4 Argument roles

Argument roles are similar to participant roles. The main difference lies in the fact that participant roles are the roles supplied by the verb, whereas argument roles are supplied by the construction.

There is also a notion of profiling with respect to argument roles. Constructional profiling occurs as follows: Every argument role linked to a direct grammatical relation is constructionally profiled.

The profiling of participant roles and the profiling of argument roles are not of the exact same kind. The criterion for determining which of a verb's participant roles are profiled is that all and only obligatory expressed participant roles are profiled. The test for which of a construction's argument roles are profiled is different. In the case of argument roles, all and only roles which are expressed as direct grammatical relations are considered profiled.

6.3 Role Linking Principles

The following are some of the principles of integration between verbs and constructions:

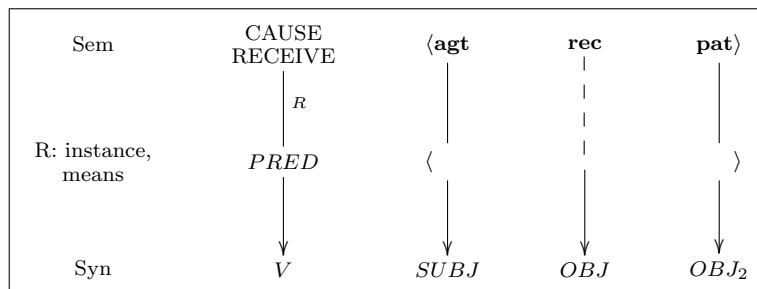
- Constructions must specify in which ways verbs will combine with them.

- They need to constraint the class of verbs that can be integrated with them.
- They must specify the way in which the event type designated by the verb is integrated into the event type designated by the construction.

The way in which verbs and constructions are combined together is by means of a fusion between argument and participant roles. Which argument roles are fused with which participant roles is determined by the two following principles:

1. The semantic coherence Principle: r_1 and r_2 are semantically compatible if r_i can be construed as an instance of r_j ($i \neq j$). Whether a role can be construed as an instance of another role is determined by general categorization principles.
2. The correspondence principle: Each participant role that is lexically profiled and expressed must be fused with a profiled argument role of the construction. If a verb has three profiled participant roles, one of them may be fused with a nonprofiled argument role of the construction.

The following example illustrates Goldberg’s representation of argument structure constructions:



The semantics associated directly with the construction is ‘CAUSE-RECEIVE <**agt pat rec**>’. PRED is a variable that is filled by the verb when a particular verb is integrated into the construction. The construction specifies which roles of the construction are obligatory fused with roles of the verb: these are indicated by a solid line between the argument roles and the verb’s participant role array. Roles which are not obligatory fused with roles of the verb —that is, roles which can be contributed by the construction— are indicated by a dashed line. The construction also specifies the way in which the verb is integrated into the construction —what type of relation R can be. From now on, profiled roles are represented by boldface letters.

There are several possible relations between the array of argument roles and the array of participant roles, some of which we will illustrate by means of the followin examples:

Perfect match

Sem	CAUSE RECEIVE	⟨ agt	rec	⟨ pat ⟩
	 R			
R: instance, means	<i>HAND</i>	⟨ hander	handee	handed ⟩
Syn	<i>V</i>	<i>SUBJ</i>	<i>OBJ</i>	<i>OBJ₂</i>

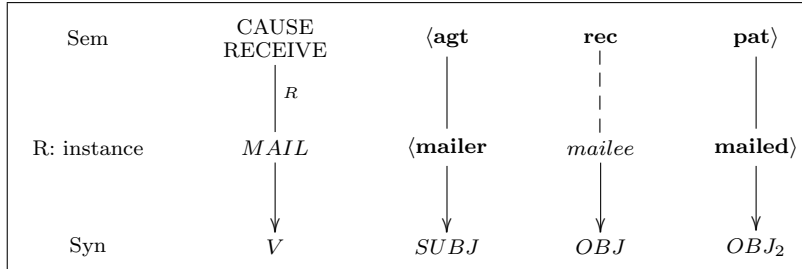
In this case, both the argument and the participant roles are all profiled and equal in number. So in this case the construction's meaning is superfluous, because the whole information is provided solely by the verb. Consider the two following cases, in which there is a profiled role fused with a non profiled role.

A non profiled argument role

Sem	CAUSE MOVE	⟨ cause	<i>goal</i>	⟨ theme ⟩
	 R			
R: instance, means	<i>PUT</i>	⟨ puter	put.place	puttee ⟩
Syn	<i>V</i>	<i>SUBJ</i>	<i>OBL</i>	<i>OBJ</i>

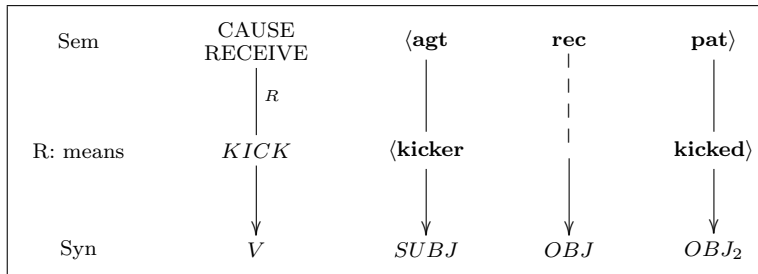
In this example, the profiled participant **put.place** role can be fused with the non profiled goal argument role, because the other two participant roles are instances of the other two argument roles. Besides, the correspondence principle asserts that if the three participant roles are profiled, one of them can be fused with one non profiled argument role. Now, the following example which represents the reciprocal situation.

A non profiled participant role



Since the correspondence principle requires only profiled participant roles be fused with profiled argument roles, and not the other way around, then the previous example is a correct case of fusion. When the argument role is profiled, then this status is inherited to the participant role. Now, the last example, with a mismatch between number of roles:

Different number of roles



This structure is produced by sentences like ‘Joe kicked Bill the ball’, where the **rec** argument is supplied by the construction.

6.4 Possible Relations Between Verbs and Constructions

Now we turn to a very important topic in the CG approach to argument structure as presented by Goldberg. Namely, the relation that should obtain between verb and construction meaning in order for them to be able to be fused.

On a constructional approach to argument structure, in which the semantics of the verb classes and the semantics of the constructions are integrated to yield the semantics of particular expressions, the question arises as to what range of verb classes can be associated with a given construction. Could *any* verb class in principle be conventionally associated with a particular construction?

The following is a list of allowed relationships between verb and construction meaning. Let e_c be the event type designated by the construction, and e_v the event type designated by the verb:

e_v **may be a subtype of e_c** . Commonly, the event type designated by the verb is an instance of the more general event type designated by the construction. For example, consider the use of ‘hand’ in

(6.10) She handed him the ball.

‘Hand’ lexically designates a type of transfer event; at the same time, transfer is the semantics associated with the ditransitive construction.

e_v **may designate the means of e_c** . Verbs which do not directly denote the meaning associated with the construction often denote the *means* by which the action is performed. Consider the following example, in which kicking is the means by which transfer is effected:

(6.11) Joe kicked Bob the ball.

e_v **may designate the result of e_c** . This follows from the ‘causal relation hypothesis’: the meaning designated by the verb and the meaning designated by the construction must be integrated via a (temporally contiguous) causal relationship. Consider the following examples:

(6.12)

- a. The wooden-legged man clumped into the room.
- b. The train screeched into the station.
- c. The fly buzzed out of the window.

Such verbs can be used freely when the sound is a *result* of the motion and occurs simultaneously with the motion. Cases which violate this are:

(6.13)

- a. *The bird chirped out of the cage.
- b. *The dog barked into the room.
- c. *The rooster crowed out of the barn.

e_v may designate a precondition of e_c . Verbs may also code particular *preconditions* associated with the semantics of the construction. For example, creation verbs designate an act of creation, which is a precondition for transfer. Consider

(6.14) Sally baked Harry a cake.

This sentence does not entail that the baking itself was causally related to the transfer. The baking does not cause the transfer, and the transfer does not cause the baking. However, the creation of the cake is a necessary precondition of the transfer.

To a very limited extent, e_v may designate the manner of e_c , the means of identifying e_c , or the intended result of e_c .

The *way* construction tends to be used with pure *manner* verbs only when the manner is particularly salient and emphasized. Consider:

(6.15)

- a. She kicked her way out of the room.
- b. "...[anyone] watching would have thought he was *scowling* his way along the fiction shelves in pursuit of a book."

The conative construction also permits exceptions to the Causal Relations Hypothesis:

(6.16)

- a. Ethel struck at Fred.
- b. Ethel shot at Fred.

In this case the verb designates the *intended result* of the act denoted by the construction.

Goldberg, following Matsumoto, introduces a further constraint in the relation between verb and construction: it is obligatory for e_c and e_v to share at least one role.

Chapter 7

CG and EC

This chapter is devoted to the formalization of CG by means of EC. But ‘formalization’ might not be the correct word here. It will rather be something like ‘first attempt at formalizing CG’. Two constructions are examined here. The ditransitive and the caused motion construction. Each of these constructions are given a scenario in EC; scenario that is later subjected to modifications, in an attempt to cope with the constraints that such constructions are subjected to. The task of producing the semantic representation of some examples, taking into account the constructions meaning, does nothing but bringing out the complexity of the task at hand. After this analysis is presented, a recent attempt at formalizing CG (Cf. [10]) is summarized, and a suggestion for the integration with the present attempt is outlined. In the final section I take stock, and briefly assess the results accomplished.

7.1 Ditransitive Construction in EC

The ditransitive construction is the following:

(7.1) X causes Y to receive Z.

The central sense is this: the agent successfully causes recipient to receive patient. In order to get to grips with this central sense it will be helpful to take a look at some examples:

(7.2) Peter gave a present to his mother,

(7.3) John threw Mary the ball,

(7.4) Mary was given a pen by John.

It could also be useful to consider verbs whose meaning closely resembles the meaning of the ditransitive construction. To begin with, we have verbs that inherently signify acts of giving, like ‘give’, ‘pass’, ‘hand’, ‘serve’, ‘feed’, etc. We also have verbs of instantaneous causation of ballistic motion, like ‘throw’,

‘toss’, ‘slap’, ‘kick’, ‘poke’, etc. Finally, we have verbs of continuous causation in a deictically specified direction, like ‘bring’ and ‘take’.

After considering the central meaning of the construction we can set about to formalize it. The leading idea of the formal representation is this. We are going to use a three-argument predicate $Of(x, z, t)$ and we are going to reify it as a fluent $Of[x, z]_f$. The intended meaning of this predicate is that of ‘belonging’. So the fluent $Of[x, z]_f$ stands for the times t at which z belongs to x . The act of giving—or to cause the transference— will be represented by two things. Firstly, we need the simultaneous change in truth value, at time t_1 say, between $Of[x, z]_f$ and $Of[y, z]_f$, where the former comes to be false and the latter comes to be true. Secondly, we will agree in saying that x successfully causes y to receive z if x causes the development of a parameterized fluent whose end point is t_1 , and which causes both the termination of $Of[x, z]_f$ and the starting of $Of[y, z]_f$.

The language in EC we need in order to formalize the latter intuition is the following:

- We need a fluent $Of[-, -]_f$ of sort σ^f ,
- We need constants a and c of sort $\sigma^{\mathbb{R}}$,
- We need an activity fluent $trans[x]_{act}$ of sort σ^{act} , and its corresponding canonical events $start$ and $finish$ of sort σ^e ,
- We also need a parameterized fluent $path_z$ of sort σ^F and a monotone increasing real function g of sort $\sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$.

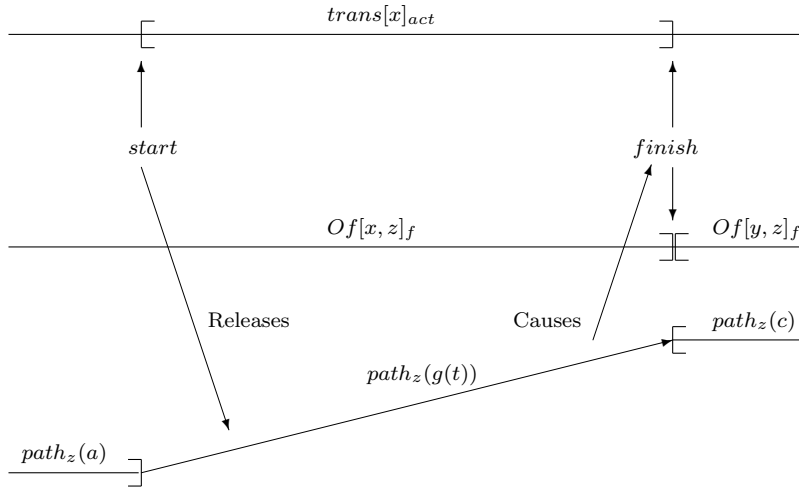


Figure 7.1: This is a visual presentation of the idea behind the formalization of the DITRANSITIVE CONSTRUCTION. The horizontal lines represent the time line. The left square brackets represent the beginning and the right square brackets represent the end of a predicate being true.

The scenario that captures the intuitive formalization presented before is this:

$$\{a < c, \text{HOLDSAT}(\text{path}_z, r_1, t) \wedge \text{HOLDSAT}(\text{path}_z, r_2, t) \rightarrow r_1 = r_2, \\ \text{INITIALLY}(\text{path}_z, a), \text{INITIALLY}(\text{Of}[x, z]_f), \text{INITIATES}(\text{start}, \text{trans}[x]_{act}, t), \\ \text{TERMINATES}(\text{finish}, \text{trans}[x]_{act}, t), \text{RELEASES}(\text{start}, \text{path}_z, r, t), \\ \text{HOLDSAT}(\text{path}_z, g(r_1), t) \rightarrow \\ \text{TRAJECTORY}(\text{trans}[x]_{act}, t, \text{path}_z, g(r_1 + r_2), r_2), \\ \text{HOLDSAT}(\text{trans}[x]_{act}, t) \wedge \text{HOLDSAT}(\text{path}_z, c, t) \rightarrow \text{HAPPENS}(\text{finish}, t), \\ \text{INITIATES}(\text{finish}, \text{Of}[y, z]_f, t), \text{TERMINATES}(\text{finish}, \text{Of}[x, z]_f, t)\}$$

With verbs whose meaning is that of ballistic motion we can use the predicate $\text{TERMINATES}(\text{start}, \text{Of}[x, z]_f, t)$ instead of $\text{TERMINATES}(\text{finish}, \text{Of}[x, z]_f, t)$ in the previous scenario.

The following constraints should also make part of the formalization of the ditransitive construction:

- There is a constraint that the transfer be caused by the agent. This follows straightforwardly from the unification of the verb and the $\text{trans}[x]_{act}$ activity, since it is that activity that causes the motion.
- The argument role z should be unified with some object to which we can ascribe a parameterized fluent —which means that the object can be ascribed a path, or an incremental change.
- We need to specify a set of constraints on the fluent $\text{Of}[x, z]_f$ that makes it a plausible characterization of the predicate ‘Belonging’.

7.2 Using EC Ditransitive Construction

I will develop, in a very informal fashion, a possible way to translate a given sentence in natural language to a semantic representation in EC, taking into account our CG approach.

Consider example (7.3), here reproduced as:

(7.5) John threw Mary the ball
 John[x_1]_f *throw*[x_4]_e *Mary*[x_2]_f *the ball*[x_3]_f

We have supposed here that we already identified the semantic representation of the lexicon. Now, we face the following problem:

1. How to match the free variables of the objects with the appropriate variables of the construction? (And how to choose the correct construction in the first place?)

Since we are following Goldberg’s proposal, the ditransitive construction is determined by the sentence type of the form $NP V NP NP$. Therefore,

the free variables of the objects and the variables of the construction are linked in a straightforward way.¹

2. How to link the verb fluent with the scenario of the construction?

We will focus on one aktionsart, namely, accomplishments. The scenario for this aktionsart provides an activity. It is this activity that we will unify with the $trans[x]_{act}$ activity of the scenario of the construction.

The outcome of the process would look like a set of unification constraints. So, for our example (7.3), the set of unifications would be this:

$$\{x = x_1; y = x_2; z = x_3; \\ trans[x]_{act} = Throw[x_4]_{act}; x = x_4\}$$

This set of constraints can be added to the scenario determined by the construction and the lexical elements. If we could work out an integrity constraint from the tense of the verb, we would have an interpretation of the sentence in EC.

Let us consider an example where the construction supplies argument roles that do not have a corresponding participant role, and let us try to work it in more detail. Consider the example:

Daniel	kicked	Tom	the ball
<i>Daniel</i> [x_1] _f	<i>kick</i> (x_4, t)	<i>Tom</i> [x_2] _f	<i>the ball</i> [x_3] _f
$x_1 = \mathbf{Daniel}$	Tense=past	$x_2 = \mathbf{Tom}$	$x_3 = \mathbf{the\ ball}$

We get a set of unification constraints between the objects and the participant roles straightforwardly from the order of the objects:

$$\{x = x_1; y = x_2; z = x_3\}$$

We need now to identify the aktionsart of the sentence. Since the sentence is *NP V NP NP*, the relevant VP is given by the verb and the second NP in the sentence. We can then look up in a database what its aktionsart is. I will assume we identified in our example the aktionsart, which is an accomplishment. We continue the representation by introducing EC constants for the accomplishment and its respective scenario. The constants consist of the predicate $kick(x_4, t)$ and the activity constructed from it: $kick[x_4]_{act}$. The unification of this activity and the activity of the construction can be introduced just as another constraint:

$$trans[x]_{act} = kick[x_4]_{act}$$

¹This question takes on a more complicated form if we were following Michaelis. For the linking between the objects of the sentence—that is, ‘John’, ‘Mary’, and ‘the ball’—and the argument roles is not straightforwardly given by their order in the sentence. For one thing, in the sentence:

(7.6) Mary was thrown the ball by John,

the link between the objects and the argument roles is not given by their order in the sentence. Furthermore, there is no principle way in which we can distinguish ‘Mary’ from ‘John’. If our criteria is purely semantic, we can make any of them the agent argument role. We will need aid from the syntax of the sentence to link objects and argument roles.

We can now build up a scenario out of the constraints on the constants for the objects, the aktionsart of the VP, the scenario for the construction, and the linking constraints. Adding the appropriate integrity constraint for the past tense of an achievement we get the full semantic representation in EC.

Finally, consider a sentence like:

Patrick sleeps George the piano
Patrick[x_1]_f *sleep*[x_4]_f *George*[x_2]_f *the piano*[x_3]_f

This sentence cannot be represented. For one thing, the verb is a state, and thus is represented by means of the fluent *sleep*[x_4]_f. This cannot be unified with the activity *trans*[x]_{act}. So the whole unification fails. This failure on unification should be enough to discard a representation of this sentence. By the same reason, no state can be represented this way. The proposal is to use this mechanism of unification to handle with any other constraints we require either on the argument roles of the construction or on the meaning of the verb.

Further constraints are needed, indeed. Consider the following example:

(7.7) *A gift was given Pat by Jennifer.

This sentence *can* be unified in the set up I have laid out so far. Since this sentence is clearly wrong, we need to specify constraints that rule it out. A constraint requiring the ‘recipient’ to be animate would certainly help us to rule (7.7) out. However, this would be too strong, since we would rule out also sentences which are grammatical, like:

(7.8) The music lent the party a festive air.

The way out taken by Goldberg is to keep the constraint on the ‘recipient’, and treat (7.8) as a metaphor. The adequacy of this solution needs to be assessed with care, since it is not clear how to treat metaphors within a formal framework.

7.3 The Caused Motion Construction in EC

The caused motion construction is this:

(7.9) X causes Y to move Z.

The basic semantics of this construction is that the causer argument directly causes the theme argument to move along a path designated by the directional phrase. Let us consider some examples:

(7.10) Frank pushed it into the box,

(7.11) Tony sneezed the tissue off the nightstand,

(7.12) George kicked the dog into the bathroom.

I will go about formalizing the meaning of this construction in the following way. I will base the analysis on a three-place predicate $In(y, z, t)$. I will reify it in order to get the fluent $In[y, z]_f$. This fluent will represent the times t at which y is in z . Here, z is intended to be a location, like ‘the box’, or the ‘nightstand’. Thus, I understand that x directly causes y to move into z if x causes the development of a parameterized fluent which ends in the true of $In[y, z]_f$. Similarly, I understand that x directly causes y to move off z if x causes the development of a parameterized fluent that ends in the true of $\overline{In}[y, z]_f$. Which one to use depends on the directional phrase.

The language in EC we need in order to formalize the latter intuition is the following:

- We need a fluent $In[-, -]_f$ and its contradictory $\overline{In}[-, -]_f$, both of sort σ^f ,
- We need constants a and c of sort $\sigma^{\mathbb{R}}$,
- We need an activity fluent $mov[y]_{act}$ of sort σ^{act} , and its corresponding canonical events $start$ and $finish$ of sort σ^e ,
- We also need a parameterized fluent $path_z$ of sort σ^F and a monotone increasing real function g of sort $\sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$.

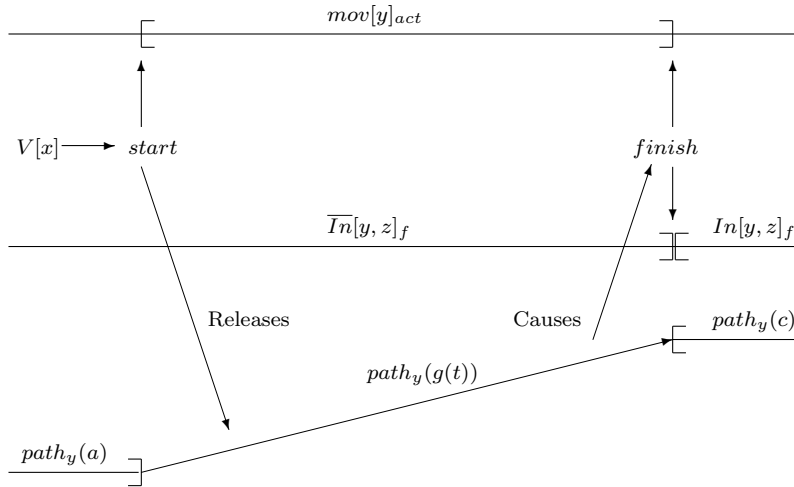


Figure 7.3: This is a visual presentation of the idea behind the formalization of the CAUSED MOTION CONSTRUCTION. The horizontal lines represent the time line. The left square brackets represent the beginning and the right square brackets represent the end of a predicate being true.

The scenario that captures the intuitive formalization presented before is this:

$$\{a < c, \text{HOLDSAT}(\text{path}_y, r, t) \wedge \text{HOLDSAT}(\text{path}_y, s, t) \rightarrow r = s, \\ \text{INITIALLY}(\text{path}_y, a), \text{INITIALLY}(\overline{In}[y, z]_f), \text{INITIATES}(\text{start}, \text{mov}[y]_{act}, t), \\ \text{TERMINATES}(\text{finish}, \text{mov}[y]_{act}, t), \text{RELEASES}(\text{start}, \text{path}_y, r, t), \\ \text{HOLDSAT}(\text{path}_y, g(r_1), t) \rightarrow \\ \text{TRAJECTORY}(\text{mov}[y]_{act}, t, \text{path}_y, g(r_1 + r_2), r_2), \\ \text{HOLDSAT}(\text{mov}[y]_{act}, t) \wedge \text{HOLDSAT}(\text{path}_y, c, t) \rightarrow \text{HAPPENS}(\text{finish}, t), \\ \text{INITIATES}(\text{finish}, In[y, z]_f, t)\}$$

If prepositions like ‘off’ occur in the directional phrase, we interchange $In[y, z]_f$ by $\overline{In}[y, z]_f$ in the previous scenario.

Another comment is in place. The verb of the sentence is not going to be unified with the activity of the construction. Rather, it is the verb that causes the activity to happen. That is what the ‘ $V[x] \rightarrow$ ’ stands for in the previous diagram. The exact details of the causation of the movement on the part of the verb need to be worked out along with the constraints on the nature of the causation of the movement that apply to the whole construction. See below for an elaboration on this point.

As it was in the case of the ditransitive construction, we have constraints on the participant roles of the caused motion construction. But, in this case, we only have a constraint on the causer role, namely, the causer can be an agent or a natural force, but it cannot be an instrument. I don’t know how to handle this using only the tools of EC.

On the other hand, Goldberg introduces five constraints on the nature of the causation represented by the Caused Motion Construction. These constraints are the following:

- I. No cognitive decision can mediate between the causing event and the entailed motion.

This constraint intends to explain the asymmetry found in the following examples:

(7.13) Sam coaxed Bob into the room.

(7.14) *Sam encouraged Bob into the room.

If Bob is coaxed, it seems that his going into the room is not mediated by a cognitive decision. If Bob is encouraged, his going into the room depends on his willingness to go into the room, and then a cognitive decision takes place between the causing event —Sam’s words of encouragement, say— and Bob’s motion.

How do we go about formalizing this constraint? We might need to characterize in some way the cognitive decisions of the theme argument role.

This is not straightforwardly achieved, but what we can do is to block y 's movement —that is, $mov[y]_{act}$ — from being caused by any of y 's plans. Let e_y be any event that is the result of any of y 's plan.² We introduce the following integrity constraint:

$$?INITIATES(e_y, mov[y]_{act}, t) \text{ FAILS}$$

In this way, we make sure that it is not up to the theme role to cause its own motion. Finally, we should specify that it is precisely the agent role that does cause the motion. We can do this by identifying the canonical event of the verb's aktionsart, and unifying it with the start event (that is, we need to introduce the constraint $e_v = mov[y]_{act}$, where e_v is the canonical event of the aktionsart).

- II. If motion is not strictly entailed, it must be presumed as a *ceteris paribus* implication.

This constraint intends to explain the asymmetry found in the following examples:

(7.15) Sam asked Bob into the room.

(7.16) *Sam begged Bob into the room.

We need to stop here for a moment and compare this constraint with the previous one. To begin with, note that in example (7.15) the theme role does make a cognitive decision in order to go into the room, and yet it is a correct sentence. Thus, there seems to be a contradiction with the first constraint. However, as Goldberg points out, Bob's going into the room is not directly caused by Sam's asking. But that asking indeed 'causes' Bob's going into the room, because that is why the sentence determines a *caused* motion construction. The causation is thus some sort of 'causation', although it may turn out that this 'causation' is not the sort of relation to which we normally attribute the term 'causation'. On the other hand, why can we ascribe an 'indirect causation' to the verb 'ask' in (7.15), and not to the verb 'beg' in (7.16)? Besides, what does it mean for the causation to be *ceteris paribus*? The answer might be this. The relation between Sam's asking and Bob's response is not a causation in the whole sense of 'causation'. However, the relation might be understood in terms of an obligation that Bob acquires when he hears Sam's request. Obligation is not a 'direct causation', but we expect Bob, *ceteris paribus*, to answer in the correct way. Yet, Bob might well ignore Sam's request and not go into the room. I believe that the tools with which EC provide us might not be the more appropriate to handle this case. The reason is that obligation is not the same as causation. There seems to be a *prima facie* difference between what someone causes and what he is obliged to do. I am not

²This kind of events are the events that come from a dynamics between an activity whose agent is y and a parameterized fluent. If I were to be rigorous, I would have to spell this out by means of the appropriate scenario.

saying that this cannot be handled *in principle* by EC. I am saying that if EC is to cope with obligations, we need to introduce more tools than we have available so far. This can be an interesting subject to be explored further, but it is out of the scope of the present work.

- III. Conventionalized scenarios can be cognitively packaged as a single event even if an intervening cause exists.

This is not a constraint, but rather a clause that allows some examples to be considered as caused motion constructions. Consider, for instance:

(7.17) The company flew her to Chicago for an interview.

This is Goldberg’s explanation of this example: ‘This is acceptable since paying for and arranging a ticket for someone else are conventional ways to have someone travel for interviews’.³

‘Packing’ many fluents into a single event is easily accomplished by EC by means of hierarchical planning (see [19, p. 94]). The event so obtained can be unified with the *start* event in the construction. On the other hand, what the fluents are, and how to decide whether these are conventional is not a task of EC.

- IV. If the activity causing the change of state effects some incidental motion and, moreover, is performed with the *intention* of causing the motion, then the path of motion may be specified.

This means that the motion of the theme role might be a side effect of the activity, but then the agent argument should have had the intention that the motion took place in the specified path. Consider the examples:

(7.18) The butcher sliced the salami onto the wax paper.

(7.19) *Pat shot Sam across the room.⁴

(7.20) *Sam unintentionally broke the eggs onto the floor.

If the motion is a side effect of the event, this should be codified somehow in the scenario of the event. That is, a clause similar to (7.21) should be part of the description of the verb:

$$\text{HAPPENS}(e_v, t) \rightarrow \text{HAPPENS}(start_m, t) \quad (7.21)$$

where e_v is the canonical event of the eventuality, and $start_m$ is the canonical event of the motion that results from the eventuality. This is a lexical information. For instance, ‘slice’ is an eventuality that determines a movement. Its representation in a scenario will include an eventuality. Call it e_v . The movement determined by this eventuality should have an initial event. Call it $start_m$.

³[5, p. 169].

⁴Unacceptable on the interpretation that Pat shot Sam and the bullet forced him across the room [5, p. 170].

To cope with the constraint in question, we can just require that the event $start_m$ be unified with the event $start$ of the construction only if the e_v is product of an intentional act. (Intentional acts are characterized in EC as the result event of a planning activity. See [19, §5.4, p. 127].)

- V. The path of motion must be completely determined by the causal force.

This constraint does not concern us much for two reasons. Firstly, the scenario of the construction completely specifies the path of motion of the theme role. This is precisely how EC works: the activity specifies the dynamics of the parameterized fluent. Secondly, this constraint has to do more with pragmatics than with semantics. This is what Goldberg says: ‘Which paths count as being “completely determined” is in part a matter of pragmatics.’⁵ Consider example (7.22):

(7.22) They laughed the poor guy into the car.

As it stands, it sounds odd (according to Goldberg). But it is completely grammatical if the path of the poor guy is specified by the context, like in the following situation. ‘[...] Imagine a group of teenagers crowded around a man who is standing by the door of his car waiting for a friend. The teenagers are intimidating the man by making jokes about him and laughing.’⁶ In cases such as (7.22), it is a business of pragmatics to provide the path of movement, if it need to be provided at all.

7.4 Using EC Caused Motion Construction

As an attempt at giving a semantic representation of an example, let us work out example (7.18). Suppose given the following representation from the lexical database:

The butcher <i>the butcher</i> [x_1] _f $x_1 = \mathbf{the\ butcher}$	sliced <i>slice</i> (x_6, t) Tense=past HAPPENS(e_{slice}, t) \rightarrow HAPPENS($start_m, t$)	the salami <i>the salami</i> [x_2] _f $x_2 = \mathbf{the\ salami}$
onto the wax paper <i>Onto</i> [x_3, x_4] <i>the wax paper</i> [x_5] _f $x_3 = \mathbf{the\ wax\ paper}$		

The unification between the objects and the argument roles is carried out by the syntax, from which we obtain the following unification constraints:

$$\{x = x_1; y = x_2; z = x_5; x_3 = x_2; x_4 = x_5\}$$

Syntax also provides us with the VP, and an appropriate database tells us that the VP is an accomplishment. This implies that ‘the salami’ determines

⁵[5, p, 173].

⁶Ibidem.

a parameterized fluent, *salami(t)*, and ‘slice’ determines a dynamics —using a monotone decreasing function— in which the salami finally disappears. Since the agent role is the subject of an activity, we can consider the slicing act as an intentional act —the slicing coincides with the preparation activity required by EC. Thus, the unification between the $start_m$ event of the verb and the $start$ event of the construction can be carried out by means of the constraint $start_m = start$. We can now collect all this constraints and the scenarios of the constraint and the aktionsart in order to build up a single scenario. The integrity constraint for the past tense of an accomplishment finishes the semantic representation in EC.

7.5 An Improvement Using Grammar

My suggestion is simple: EC is not alone in the fight. We might need a whole handful of allies in order to tackle these very difficult problems. (In §5.3, I gave a suggestion that points out why EC might not be enough.) My proposal in this work is just to show that EC might be helpful in the coalition that seeks to formalize CG. The first part of the proposal is that it is helpful because it can be used to provide (part of the) meaning of the constructions, and we can derive some constraints from this meaning.⁷ The second part of the proposal is to show that it can be integrated with other members of the coalition. For this part, I use Kay’s proposal [10] of formalization of CG. In this section, I will first give a brief summary of this⁸, and then I will suggest how the integration between it and EC could be carried out.

7.5.1 Kay’s attempt at formalizing CG

Kay’s proposal is given within a Constraint-based framework. Therefore, we need to make the distinction between the domain of formal objects among which the grammar licences the constructs of the language and the feature descriptions, which license the constructs of the grammar. In other words, the domain of formal objects is a big set, and not all formal objects are licensed by the language. We need, then, to specify a set of feature descriptions and a relation of satisfaction between feature descriptions and possible constructions. The language will be defined as the set of formal objects that are licensed by at least one feature description.

⁷Two clear contributions are the ruling out of stative verbs in the ditransitive construction, and the integrity constraint that copes with the first constraint of the caused motion construction. I am sure more can be done.

⁸I will not be one hundred per cent faithful to Kay’s presentation, mainly because I think it is not very neat from a mathematical perspective. I will rather try to improve on the notation, but I will not go into all the details of his proposal.

Feature Structure Trees

The formal objects are *Feature Structure Trees*. In order to define them, we need first to define what the *Feature Structures* are. In order to do so, we need a finite family of finite sets $A_0, A_1, A_2, \dots, A_k$ of *attributes* and a set V of *values*, together with a function $\varphi : A_0 \rightarrow \wp(V)$. By using these we define the *features*:

Definition (Features). 1. A feature of level zero is a pair $\langle a, v \rangle$ where $a \in A_0$ and $v \in \varphi(a) \subseteq V$. The set of features of level zero is called \mathfrak{F}_0 .

2. A feature of level $n+1$ is a pair $\langle a, v \rangle$ where $a \in A_{n+1}$ and $v \in \wp(\bigcup_{i=0}^n \mathfrak{F}_i)$. There is a requirement on v . If $\langle a', v'_1 \rangle, \langle a', v'_2 \rangle \in v$, then $v'_1 = v'_2$. The set of features of level $n+1$ is called \mathfrak{F}_{n+1} .

Examples of features are the following. Suppose $A_0 = \{num, per\}$, $A_1 = \{agr\}$, $V = \{1st, 2nd, 3rd, sg, pl\}$, $\varphi(num) = \{sg, pl\}$, and $\varphi(per) = \{1st, 2nd, 3rd\}$. Features of level zero are, for instance, $\langle per, 3rd \rangle$, or $\langle num, pl \rangle$. A feature of level one is, for instance, $\langle agr, \{\langle per, 3rd \rangle, \langle num, pl \rangle\} \rangle$.

Definition (Feature Structure). A Feature Structure is a subset F of $\bigcup_{n=0}^k \mathfrak{F}_n$ such that for every $a \in \bigcup_{n=0}^k A_n$, if $\langle a, v_1 \rangle, \langle a, v_2 \rangle \in F$, then $v_1 = v_2$. The set of all Feature Structures will be called \mathfrak{FS} .

Now we can give the definition of the formal objects. These formal objects have a proper name: Feature Structure Trees.

Definition (Feature Structure Trees). A structure $\langle N, <, M, \phi \rangle$ is a Feature Structure Tree if:

- $\langle N, < \rangle$ is a strict partial ordered set (the set N is thought of as a set of nodes),
- M is a partial function from N to N (M is the ‘mother of’ function),
- ϕ is a partial function from N to FS .

and the following conditions hold:

1. Exactly one node $r \in N$ has no image by M (i.e., $M(r)$ is undefined). This node is called ‘the root’,
2. For distinct nodes $n_i, n_j \in N$, if $M(n_i) = M(n_j)$, then $n_i < n_j$ or $n_j < n_i$,
3. For every $n_i, n_j \in N$, if $M(n_i) < M(n_j)$, then $n_i < n_j$.
4. For every $n \neq r$ (with r the root) there is a $k \in \mathbb{N}$ such that $M^{(k)}(n) = \underbrace{M \circ \dots \circ M}_{k\text{-times}}(n) = r$.
5. For every $k \in \mathbb{N}$, it is not the case that $M^{(k)}(n) < n$ or $n < M^{(k)}(n)$.

Feature Structure Trees, or FT’s for short, can be defined by means of equations. Two different types of equations should be considered. We have CS equations, that define the structure of the tree, and FS equations, that define the function ϕ that assigns Feature Structures to the nodes of the tree.

CS Equations

For motivation, consider the following set of equations:

$$\begin{aligned} M(n_2) &= n_1 & (7.23) \\ M(n_3) &= n_1 \\ n_2 &< n_3 \end{aligned}$$

This set of equations define the following tree (the M function specifies the edges and the $<$ relation specifies the left-to-right order of the nodes):



A set of equations that satisfies the requirements in the definition of FT defines a tree. This is intuitive enough so we do not need to go into details.

FS Equations

Let $F \in \mathfrak{FS}$ and $a \in \bigcup_{n=0}^k A_n$. Note that, by definition of F , if there is a v such that $\langle a, v \rangle \in F$, then v is unique. This allows us to think of F as a partial function with domain $\bigcup_{n=0}^k A_n$. That is, $F(a) = v$ iff $\langle a, v \rangle \in F$. Suppose $\langle a, v \rangle$ is a feature of level $n + 1$. Then v is itself a set of pairs of features of level $\leq n$, where no two pairs share the same first component. So v is a partial function with domain $\bigcup_{i=0}^n A_i$. If $a' \in A_i$, then we define $v(a') = v'$ iff $\langle a', v' \rangle \in v$. This process can be continued down to the level zero.

Let FT be a FT. We have a partial function ϕ from N to \mathfrak{FS} . For each n in the domain of ϕ , $\phi(n)$ is a feature structure. Using the comments on the previous paragraph, we can specify any such feature structure by a set of equations.

For motivation, take the feature structure:

$$\phi(n) = \{\langle phon, \langle walk \rangle \rangle, \langle agr, \{\langle per, 3rd \rangle, \langle num, pl \rangle\} \rangle\} \quad (7.24)$$

(7.24) can be specified by the following set of equations:

$$\begin{aligned} \phi(n)(arg)(per) &= 3rd & (7.25) \\ \phi(n)(arg)(num) &= pl \\ \phi(n)(phon) &= \langle walk \rangle \end{aligned}$$

Descriptions

Note that a FT can be specified —and underspecified— using equations of the form (7.23) and (7.25). Such sets of equations will be called *descriptions*. We say that a description is *satisfied* in a set L of FT's iff there is a FT in L that satisfies the equations of the descriptions. If a description is satisfied by a FT, we say that the description *licenses* that FT.

We can understand now the idea of the previous formalization. We should choose appropriate sets A_0, \dots, A_k, V and an appropriate set \mathcal{G} of descriptions such that every FT which represents a sentence in the language (English, say) is licensed by some combination of descriptions. Furthermore, every combination of descriptions that should license an FT licenses a representation of a sentence in the language.

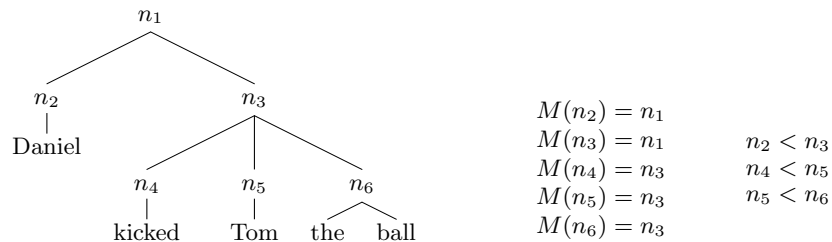
7.5.2 Integration of Kay's attempt with EC

My suggestion of integration comes in the form of an example. Consider example:

(7.26) Daniel kicked Tom the ball.

In order to license this example we need to provide features out of which we can build up appropriate FT's. This task is out of the scope of the present work, and I will just bring out some important points. I am particularly interested in arguing that the link between objects and participant roles can be provided within this combined approach.

To begin with, note that a set of constraints can be given in order to define the tree of the sentence:



A set of equations can be given in order to define the FS assigned to each node. The following is a very incomplete list:

$$\phi(n_1)(SEM)(tense) = past$$

$$\phi(n_2)(phon) = \langle daniel \rangle$$

$$\phi(n_2)(SEM)(EC)(fluent) = Daniel[x_1]_f$$

$$\phi(n_2)(SEM)(EC)(variable) = x_1$$

$$\phi(n_2)(SEM)(EC)(scenario) = \{x_1 = \mathbf{Daniel}\}$$

$$\phi(n_3)(SYN)(HEAD) = verb$$

$$\phi(n_3)(SYN)(val)(spr)(SEM)(index) = x_1$$

$$\phi(n_3)(sem)(frame)(agent) = x_1$$

$$\phi(n_3)(SEM)(EC)(scenario) = \{.. \}$$

$$\phi(n_4)(phon) = \langle kicked \rangle$$

$$\phi(n_4)(SEM)(EC)(predicate) = kick(x_4, t)$$

Two things are worth noting. Firstly, the leafs can be assigned the semantics given in EC simply by specifying the appropriate features. Secondly, the node n_3 has information that links the subject with the agent participant role and the subject of the verb.

None of these foregoing constraints were explained, because I have not yet explained the features used therein. Unfortunately, I am not in a position to explain them. Though, what is important is the suggestion that EC representation can be reproduced in this framework by means of features, and that the node n_3 can be given the representation of the ditransitive construction. We can use a representation of this construction such as figure 7.5.2, as given in [13] (where I do not explain anyone of the features used therein either). Maybe with these tools together we will be able to work out a coherent, but formal, account of the licensing of sentences in the CG.

7.6 Comments on CG and EC

I want to finish this chapter with a sentence in Spanish:

*Todavía hay mucha tela para cortar.*⁹

This means that we have still got a lot of work to do. Indeed, I believe that nothing in this chapter can be considered as close to completion. However, that was not the objective of this part of the thesis. The objective was rather to

⁹The literal translation is something like ‘there is still a lot of cloth to cut off.’

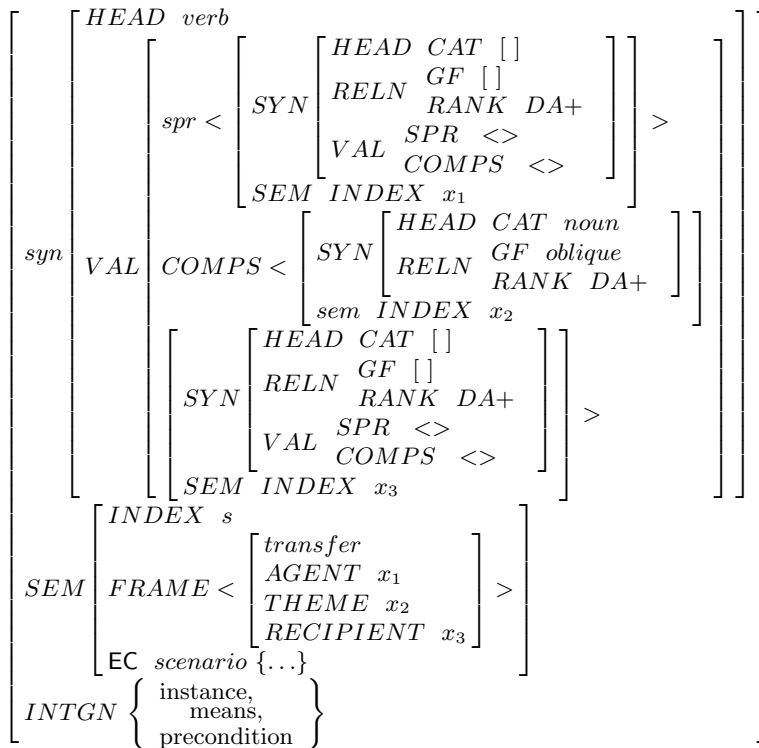


Figure 7.a: Ditransitive Construction adapted from [13], and extended with features ranging on EC.

explore the possibility of giving EC a place in CG. The advantage of such an approach, in case it can be worked out, is double. On the one hand, we will give a formal way to go from natural language to EC. On the other hand, we will give CG excellent formal tools with which we can formalize the construction meaning.

I want to assess briefly the work done in this section. To do so, I will give two lists. One list contains the positive results, i.e., these things that we gained by following the present path. The other list contains things that need to be improved, and these points on which we will need to use tools not provided by EC.

Positive things: Some good results were achieved.

1. Once the meaning of the constructions we dealt with in this chapter is identified, its coarse representation in EC is almost straightforward. This might suggest that EC provides an appropriate framework for the formalization of CG.
2. The language of EC is rich enough to characterize a number of constraints on the constructions that otherwise would be treated as primitives. For instance, the ‘agentivity’ of the agent participant role — tackled by means of the behavior of the activity fluents in EC—, the division between events and fluents —which was helpful in the ditransitive construction—, and the characterization of mediating cognitive causes —used in the caused motion construction.
3. EC language is rich enough to allow us to take a glance at the end point of the representation: a scenario full of constraints and dynamics.

To improve: A lot of things need to be worked out.

1. The meaning of the constructions may not be altogether represented in EC. Some examples are provided by the ‘belonging’ predicate — which seems to be primitive, and with no clear relation with causation—, the same goes for constraints that involve obligations instead of causes —like in the caused motion construction—, and for constraints that involve conventions and pragmatics.
2. It is not clear how we get the aktionsart for the VP. As it happened in the first part, this link comes out of the blue, and is rather supposed than analyzed.
3. The link between EC and other attempts at formalizing CG, like Kay’s, need to be elaborated in detail. Here, I barely scratched the surface.
4. The representation of other argument structure constructions need to be worked out.

Part III

A Brief Discussion

Chapter 8

Cognitive Models and Truth Conditions

8.1 Cognitive Models and Semantics

EC main theses are (i) the linguistic encoding of temporal notions can be studied with the help of the notions of planning and causality; (ii) understanding a piece of discourse involves constructing a cognitive model of that discourse.

A linguistic analysis in EC starts off by bringing out a linguistic phenomenon involving temporal notions, say the distinction between sentences (8.1-8.3):

(8.1) Caroline flies to Chicago tomorrow,

(8.2) Caroline is going to fly to Chicago tomorrow,

(8.3) Caroline will fly to Chicago tomorrow,

Accordingly, the phenomenon is explained in terms of causality and planning. For instance, the distinction in question is explained like this: (8.1) is a sort of ‘natural consequence’ of Caroline’s current situation —no plan involved here—, (8.2) expresses a plan that Caroline has got —which might be called off, even by Caroline—, and (8.3) expresses a plan that Caroline should not call off.

According to (ii), since understanding is explained in terms of a cognitive model, a cognitive model of these different causes and planes should be given. This is delivered in the form of minimal models for appropriate scenarios and integrity constraints. Given a scenario, its minimal models (usually) can be ‘computed’¹ and, for this reason, minimal models are intended to be cognitive representations.

It is important to bear in mind that tenet (i) is independent from tenet (ii). An analysis of the temporal notions and its linguistic encoding need not to be

¹This means that a formula ϕ is true in a minimal model for the scenario if and only if it can be derived in Constraint Logic Programming.

given in terms of cognitive models. Prima facie, it is possible to elaborate on the aforementioned intuitive description of the meaning of temporal discourse. And, if a more systematic explanation is sought, we could try out an explanation based on truth conditions.

Leaving this possibility open seems important to me. Mainly, because the notion of a cognitive model is somewhat obscure, and I find it difficult to accept that a cognitive model explains meaning. Let us discuss tenet (ii) in a bit more detail. It can be paraphrased in the following way. A person's understanding of a piece of discourse consists, among other things, in the construction of a cognitive model of that discourse. When a person hears/reads a piece of discourse, a cognitive model comes up to his mind, and it is this model in which his understanding of that piece of discourse consists.

My qualms with cognitive models basically refer to the possibility of judging, or testing them. How can we know that a cognitive model in fact captures, or represents the meaning of a discourse? When can a cognitive model fail to represent the meaning of a discourse? Can two different models represent the meaning of the same discourse? Can two persons have a different model representing the same discourse? How do we know that? If we want to cope with the communicability of thoughts, we assume that a communication is successful if the hearer decodes from an utterance the same cognitive model that the speaker codified in that utterance. So we assume that two persons should have the same cognitive model of the same discourse in order for communication to be successful. But how can we test that a given communication thus defined was successful? We surely cannot scan people's minds and compare their cognitive models. It can be argued that we could provide indirect evidence that they have the same cognitive model. The problem here is what counts as 'indirect evidence' that someone has a particular cognitive model. Brain activity at most gives insight as to what processes are likely to be carried out in the brain, but it cannot be brought to bear in order to distinguish two models that use the same computational process. Maybe behavioral clues in the form of systematic answers to verbal stimulus, or in the form of social dynamics? But again, how do we know that these evidence is evidence for a particular cognitive model? Besides, why can't this 'evidence' be rather taken as what we are looking for instead of an intermediate step for an obscure cognitive model? Moreover, if we had required that these persons were to have a different cognitive model, we would have assumed so to begin with. There is no way for us to be wrong about these assumptions. The problem is not that we make one of these assumptions. The problem is that we cannot falsify them. Likewise, if we have a cognitive model for a given discourse, how can we say that it is not the appropriate cognitive model for it? If we have a sentence like (8.4):

(8.4) John kicked Mary the ball.

what is the appropriate cognitive model for it? Is it the dynamic of the path described by the ball going from John to Mary? Is it just the instantaneous action happening at the time John touches the ball with his foot with the intention to send the ball to Mary? Is it the aforementioned dynamic preceded

by a dynamic describing John’s plan—in the form of another dynamic? Or does it consist of all of these dynamics? How can we decide this? By introspection? What do I ‘see’, or imagine, when I read the sentence? I suppose each person has a different ‘mental image’ of it. Which one is correct? The most frequent? Why would any of them be the wrong ‘mental image’? On the other hand, although there could be many ‘mental images’ of (8.4), it has only one meaning. There is only one way to get the meaning of it correctly. But we do not have a way to say that there is only one correct cognitive model that represents the sentence. Suppose we skip this impasse, that is, suppose that we have a way to decide which model is the correct one. Thus, each discourse is represented only by one cognitive model. Is this model pre-wired in our minds or is it learned? How do a child learn the meaning of (8.4), if it learns it at all? Certainly he cannot learn it by ‘seeing’ the cognitive model his teacher has in his mind when he utters that sentence. To be honest, I can see no use for a cognitive model when teaching the meaning of a sentence. One can say that the appropriate cognitive model shows up in the child’s mind when he has learned the meaning of (8.4), but the cognitive model was of no use in the learning process. Furthermore, how can the teacher corroborate that the child did learn the correct meaning of the sentence in question?

A cognitive model is useless for teaching the meaning of a sentence, is useless for testing whether someone correctly learned the meaning of a sentence, is useless for testing whether a communication was successful. Thus, what is it good for? There might be an answer to it. Cognitive models could render truth conditions—because they are ‘models’, not because they are ‘cognitive’—, and truth conditions are better candidates to give insight into the meaning of a sentence. I will devote now my attention to a discussion of the truth conditions given by EC.

8.2 EC Truth Conditions

This is not the place to argue that truth conditions are better than cognitive models. It is just the place to show that EC can deliver truth conditions. In order to do so, I will give a brief overview of EC formal tools that end up in the definition of truth conditions given in [19, p. 105]. Next it follows a discussion of that definition and an improvement of it is proposed. Some closing comments complete this chapter.

8.2.1 Formal tools

Formal Language The first step of EC is the definition of a sufficiently rich formal language. This language is far too complicated to be explained in this section.² It is enough to point out that it takes primitive constants of objects, events, fluents (situations that evolve though time), and real-time coordinates.

²An explicit definition of it can be found in the appendix (Cf. §9).

It also contains predicates that regulate the causal interaction between event and fluents.

Scenarios and lexical specification A logic program can be given in order to restrict the behavior of particular objects, events, and fluents. This restriction is intended to make explicit —as much as causal interactions allow us— the specification of lexical items in the language. The logic program that achieves this is called a scenario.

Semantics The previous language is a First-Order Multi-Sorted Logic. However, the logic we intend to realize is non-monotonic. Therefore, we use a class of partial Multi-Sorted models as a semantics for the formal language. This class is not straightforwardly defined, but the idea is that the models of this class are minimal, in the sense that nothing that has not been specified occurs. A minimal model for a scenario can be defined as the smallest model of the *completion* of the scenario (Cf. [19, p. 55]). The completion of a scenario \mathbf{S} will be denoted by $\bar{\mathbf{S}}$.

Syntactic deductions The syntactic counterpart of the semantics just defined is Constraint Logic Programming with negation as failure. It can be shown that this set of rules is sound with respect to the class of minimal models we chose as our semantics. It is not complete, unfortunately, as there are derivations in Constraint Logic Programming that might loop.

Integrity constraints Given a scenario \mathbf{S} and a formula ϕ which is consistent³ with \mathbf{S} , it is possible to define a ‘minimal’ success update, $\mathbf{S}_{\text{SUCCEEDS}}$, of \mathbf{S} in such a way that $\mathbf{S}_{\text{SUCCEEDS}} \supseteq \mathbf{S}$ and ϕ is true in any minimal model of $\mathbf{S}_{\text{SUCCEEDS}}$. It is also possible to define a ‘minimal’ failure update $\mathbf{S}_{\text{FAILS}}$ in such a way that $\mathbf{S}_{\text{FAILS}} \supseteq \mathbf{S}$ and ϕ is false in any minimal model of $\mathbf{S}_{\text{FAILS}}$. An Integrity Constraint is defined to be a formula ϕ along with a request to preform either a success or a failure update of \mathbf{S} . This will be denoted by $?\phi$, and the update of \mathbf{S} according to $?\phi$, when successful, will be denoted by \mathbf{S}_ϕ .

Truth conditions The language we have defined thus far already contains truth conditions. But it is important to point out that EC does not define the truth conditions of a natural language sentence \mathcal{S} directly by using the language’s truth conditions. This is due to the fact that, unlike say Montague Grammar, we do not translate \mathcal{S} into a formula in the formal language. The translation of \mathcal{S} consists of a scenario \mathbf{S} and an Integrity Constraint $?\phi$. The truth conditions of \mathcal{S} are defined as follows. \mathcal{S} is true iff the update of \mathbf{S} requested by $?\phi$ is successful.⁴

³Consistence in this case is relative to any classical model, not a minimal model. Otherwise, it would not make any sense to ask for a successful update of \mathbf{S} .

⁴Since certain derivations might loop, this ‘iff’ is too strong. It should be broken down into two clauses that define when a sentence is true and when it is false. Cf. [19], p. 105. For

8.2.2 Truth

Suppose \mathcal{S} is represented in EC by \mathbf{S} and $?\phi$, and consider the following claim:

$$\mathcal{S} \text{ is true iff the update of } \mathbf{S} \text{ with } ?\phi \text{ is successful} \quad (8.5)$$

Definition (8.5) has at least two problems. To begin with, it does not assert what \mathcal{S} is true of. In other words, it does not say against what we compare \mathcal{S} in order to decide whether it is true. It is not a remedy to replace the left-hand side of it with ‘ \mathcal{S} is true in a given model \mathfrak{A} ’. The reason being that the right-hand side does not make use of \mathfrak{A} . The second problem, derived from the first one, is that under this definition we cannot work out a coherent definition of falsity. Let me explain a bit. A sentence is false if it is not true —intuitionistic worries aside. When is \mathcal{S} not true under definition (8.5)? This depends on whether the request of update, i.e. $?\phi$, is a success or failure request. Suppose without loss of generality that it is a request of success update. This means that \mathcal{S} is false when $?\phi$ cannot update \mathbf{S} in order for ϕ to be true in any minimal model of \mathbf{S} . This only happens when \mathbf{S} and ϕ are inconsistent. But since only one \mathbf{S} and only one ϕ are assigned to each \mathcal{S} regardless its actual context, then \mathcal{S} is always true or always false. This just does not make any sense —except for tautologies or contradictions— for we need to make place for contingencies in any definition of truth conditions.

Tarski used the notion of translation into a metalanguage in order to define truth conditions. The conditions ‘ $P(a)$ is true iff $a^{\mathfrak{A}} \in P^{\mathfrak{A}}$ ’ is a clear example of this: $a^{\mathfrak{A}} \in P^{\mathfrak{A}}$ is a translation of $P(a)$ in the metalanguage (given a model \mathfrak{A}). A compositional definition of truth conditions is the key to give truth conditions to a language specified by a grammar. This is the ‘traditional way’ in semantics, so to speak. But, prima facie, we cannot follow this path in order to provide EC with truth conditions. We would need a more indirect way to do so. We can, instead, follow DRT’s proposal. That is, a sentence \mathcal{S} determines a DRS K . It is possible to define a model, \mathfrak{B}_K , from K . Given any model \mathfrak{A} , we say that \mathcal{S} is true in \mathfrak{A} iff \mathfrak{B}_K is a submodel of \mathfrak{A} .⁵ We can suggest a similar definition for EC. A sentence \mathcal{S} determines what I will call a semantic representation, that is, a \mathbf{S} and a $?\phi$. Updating \mathbf{S} we obtain \mathbf{S}_ϕ , and to it we can give a minimal model, $\mathfrak{M}_\mathcal{S}$. Given a model \mathfrak{A} , we say \mathcal{S} is true in \mathfrak{A} iff $\mathfrak{M}_\mathcal{S}$ is a submodel of \mathfrak{A} .

A closing comment is in place. It is worth mentioning that in order to make explicit the foregoing definition of truth conditions, we need to give a semantic representation to each sentence in a systematic way. Such systematic assignment of semantic representations assumes —for several reasons— a syntax specified beforehand.⁶ This leads us into either the first or the second position with respect to syntax and semantics (see §5.3). Thus, my critique of cognitive models

ease of presentation I will omit this technical detail.

⁵Truth conditions for DRT can be also defined in a more traditional way. Cf. [4, Ch. 6, Vol. 2].

⁶Some of the reasons are the dealing with syntactic ambiguity, and the possibility of giving representations to infinitely many sentences by finitistic means.

and my interest in truth conditions preclude me to agree with the approach worked on in the second part of this thesis, and suggests that improving on the first part of the thesis —either insisting on DRT or not— might be a more rewarding enterprise.

Chapter 9

Appendix — Some details of EC

9.1 The Language of EC

The language of EC is a 5-sorted First-Order language. The sorts are the following:

1. $\sigma^{\mathbb{R}}$ is the sort of reals,
2. σ^e is the sort of events,
3. σ^f is the sort of non-activity fluents,
4. σ^{act} is the sort of activity fluents.
5. σ^F is the sort of parameterized fluents.

Note that in this version there is no sort for objects. These are going to be codified as a finite list of constants of sort $\sigma^{\mathbb{R}}$ (see below).

The following predicates make part of the language of EC:

1. All the usual predicates of $L(\mathbb{R})$, particularly, the binary predicate \leq . If R is a n -ary predicate in $L(\mathbb{R})$, then it is of sort $\underbrace{\sigma^{\mathbb{R}} \times \dots \times \sigma^{\mathbb{R}}}_{n\text{-times}}$,
2. For each $n \geq 1$, a countably infinite set of constants of n -ary predicates of sort $\underbrace{\sigma^{\mathbb{R}} \times \dots \times \sigma^{\mathbb{R}}}_{n\text{-times}}$. We mentioned this in a separate item, since these constants are intended to be used to codify VP's that are to be transformed into eventualities (Cf. §3.2).
3. A binary predicate HAPPENS of sort $\sigma^e \times \sigma^{\mathbb{R}}$,
4. A 5-nary predicate TRAJECTORY of sort $\sigma^{\text{act}} \times \sigma^{\mathbb{R}} \times \sigma^F \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$,

5. A ternary predicate $\text{RELEASES}^{\sigma^f}$ of sort $\sigma^e \times \sigma^f \times \sigma^{\mathbb{R}}$.

For $\sigma \in \{\sigma^f, \sigma^{\text{act}}\}$:

6. An unary predicate INITIALLY^σ of sort σ ,
7. A ternary predicate INITIATES^σ of sort $\sigma^e \times \sigma \times \sigma^{\mathbb{R}}$,
8. A ternary predicate TERMINATES^σ of sort $\sigma^e \times \sigma \times \sigma^{\mathbb{R}}$,
9. A ternary predicate CLIPPED^σ of sort $\sigma^{\mathbb{R}} \times \sigma \times \sigma^{\mathbb{R}}$,
10. A binary predicate HOLDSAT^σ of sort $\sigma \times \sigma^{\mathbb{R}}$.

For the parameterized fluents we require a ‘two-dimensional’ variety of the previous predicates:

11. An binary predicate $\text{INITIALLY}^{\sigma^F}$ of sort $\sigma^F \times \sigma^{\mathbb{R}}$,
12. A 4-ary predicate $\text{INITIATES}^{\sigma^F}$ of sort $\sigma^e \times \sigma^F \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$,
13. A 4-ary predicate $\text{TERMINATES}^{\sigma^F}$ of sort $\sigma^e \times \sigma^F \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$,
14. A 4-ary predicate $\text{RELEASES}^{\sigma^F}$ of sort $\sigma^e \times \sigma^F \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$.
15. A 4-ary predicate $\text{CLIPPED}^{\sigma^F}$ of sort $\sigma^{\mathbb{R}} \times \sigma^F \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$,
16. A ternary predicate $\text{HOLDSAT}^{\sigma^F}$ of sort $\sigma \times \sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$.

EC uses the functions $+$ and \cdot both of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$. We need a binary function Π of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$, and functions π_1 and π_2 of sort $\sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$.

Let $\sigma \in \{\sigma^e, \sigma^f, \sigma^{\text{act}}, \sigma^F, \sigma^{\mathbb{R}}, \sigma^{\mathbb{N}}\}$. We assume a fixed, countably infinite set of variables $x^\sigma, y^\sigma, z^\sigma, \dots$ of sort σ . We assume a constant $\mathbf{0}^{\sigma^{\mathbb{R}}}$ of sort $\sigma^{\mathbb{R}}$, and a finite list of constants of sort $\sigma^{\mathbb{R}}$. This list is intended to be a codification of the objects to be used in the semantic representations.

The terms of the language of EC are built up inductively as follows:

1. Each variable of sort σ is a term of sort σ ,
2. Each constant of sort σ is a term of sort σ .
3. If g is a n -ary function of sort $\sigma_1 \times \dots \times \sigma_n \rightarrow \sigma_{n+1}$ and t_1, \dots, t_n are terms of sort $\sigma_1, \dots, \sigma_n$ respectively, then $g(t_1, \dots, t_n)$ is a term of sort σ_{n+1} .

The atomic formulas are defined as follows:

1. If t_1, t_2 are terms of sort σ , then $t_1 = t_2$ is an atomic formula,

2. If R is a n -ary predicate of sort $\sigma_1 \times \dots \times \sigma_n$ and t_1, \dots, t_n are terms of sort $\sigma_1, \dots, \sigma_n$ respectively, then Rt_1, \dots, t_n is an atomic formula.

The formulas of EC are the following:

1. Every atomic formula is a formula,
2. If φ is a formula so is $\neg\varphi$,
3. If φ and ψ are formulas so is $\varphi \wedge \psi$,
4. If x^σ is a variable of sort σ , and if φ is a formula, then $\forall x^\sigma \varphi$ and $\exists x^\sigma \varphi$ are formulas.

9.2 The Axioms of EC

In the first place, we need all the axioms of the real numbers¹ for the functions $+$ and \cdot of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}} \rightarrow \sigma^{\mathbb{R}}$, and the binary relation \leq of sort $\sigma^{\mathbb{R}} \times \sigma^{\mathbb{R}}$. Now, we add to this set of axioms the following list of axioms, where e is a variable of sort σ^e , f is a variables of sort σ^f , \mathbf{f} is a variable of sort σ^{act} , F is a variable of sort σ^F , and r, t, t', d, g are variables of sort $\sigma^{\mathbb{R}}$:

Axiom 1.1: $\text{INITIALLY}^{\sigma^f}(f) \rightarrow \text{HOLDSAT}^{\sigma^f}(f, \underline{0})$

Axiom 2.1: $(\text{HOLDSAT}^{\sigma^f}(f, r) \wedge (r < t) \wedge \neg \exists s < r (\text{HOLDSAT}^{\sigma^f}(f, s)) \wedge \neg \text{CLIPPED}^{\sigma^f}(r, f, t)) \rightarrow \text{HOLDSAT}^{\sigma^f}(f, t)$

Axiom 3.1: $(\text{HAPPENS}(e, t) \wedge \text{INITIATES}^{\sigma^f}(e, f, t) \wedge (t < t') \wedge \neg \text{CLIPPED}^{\sigma^f}(t, f, t')) \rightarrow \text{HOLDSAT}^{\sigma^f}(f, t')$

Axiom 4: $(\text{HAPPENS}(e, t) \wedge \text{INITIATES}^{\sigma^{\text{act}}}(e, \mathbf{f}, t) \wedge (t < t') \wedge (t' = t + d) \wedge \text{TRAJECTORY}(\mathbf{f}, t, F, g, d) \wedge \neg \text{CLIPPED}^{\sigma^{\text{act}}}(t, \mathbf{f}, t')) \rightarrow \text{HOLDSAT}^{\sigma^F}(F, g, t')$

Axiom 5.1: $(\text{HAPPENS}(e, s) \wedge (t < s < t') \wedge (\text{TERMINATES}^{\sigma^f}(e, f, s) \vee \text{RELEASES}^{\sigma^f}(e, f, s))) \rightarrow \text{CLIPPED}^{\sigma^f}(t, f, t')$

We will also add a set of axioms **i.2**, which are the same as axioms **i.1** (for **i = 1, 2, 3**) but with predicates and variables of sort σ^{act} instead of σ^f . We also add the following set of axioms for parameterized fluents (note that g plays the role of the parameter for F):

Axiom 1.3: $\text{INITIALLY}^{\sigma^F}(F, g) \rightarrow \text{HOLDSAT}^{\sigma^F}(F, g, \underline{0})$

Axiom 2.3: $(\text{HOLDSAT}^{\sigma^F}(F, g, r) \wedge (r < t) \wedge \neg \exists s < r (\text{HOLDSAT}^{\sigma^F}(F, g, s)) \wedge \neg \text{CLIPPED}^{\sigma^F}(r, F, g, t)) \rightarrow \text{HOLDSAT}^{\sigma^F}(F, g, t)$

¹Those of a complete order field.

Axiom 3.3: $(\text{HAPPENS}(e, t) \wedge \text{INITIATES}^{\sigma^F}(e, F, g, t) \wedge (t < t'))$
 $\wedge \neg \text{CLIPPED}^{\sigma^F}(t, F, g, t')) \rightarrow \text{HOLDSAT}^{\sigma^F}(F, g, t')$

Axiom 5.3: $(\text{HAPPENS}(e, s) \wedge (t < s < t') \wedge (\text{TERMINATES}^{\sigma^F}(e, F, g, s)$
 $\vee \text{RELEASES}^{\sigma^F}(e, F, g, s))) \rightarrow \text{CLIPPED}^{\sigma^F}(t, F, g, t')$

We need axioms to control the behavior of functions Π , π_1 and π_2 , since we want them to behave as ‘tuple coding’ function and its respective projection functions. The axioms are:

1. $\pi_1 \circ \Pi(x, y) = x$, and
2. $\pi_2 \circ \Pi(x, y) = y$.

We define the code $\Pi^n(x_1, \dots, x_n)$ of a n -tuple (x_1, \dots, x_n) by induction on $n \geq 2$:

1. $\Pi^2(x_1, x_2) = \Pi(x_1, x_2)$,
2. $\Pi^n(x_1, \dots, x_n) = \Pi(\Pi^{n-1}(x_1, \dots, x_{n-1}), x_n)$.

The projection functions for Π^n are defined by $\pi_i^n = \pi_2 \circ (\pi_1)^{n-i}$ (for $i = 1, \dots, n$), and it can be easily proven that $\text{EC} \vdash \pi_i^n \circ \Pi^n(x_1, \dots, x_n) = x_i$.

We define the numerals on each sort by induction (though we will omit the superscript referring to the sort):

1. $\bar{0} = \mathbf{0}$,
2. $\overline{n+1} = \Pi(\bar{n}, \bar{0})$.

It can be easily proven that if $\bar{n} = \bar{m}$ then $n = m$, and if $\overline{n+1} = \overline{m+1}$ then $\bar{n} = \bar{m}$.

Because of the particularities of our version of Feferman coding, we code formulas in the $\sigma^{\mathbb{R}}$ ‘realm’. On the other hand, we have a many-sorted language of events and fluents (recall that we have three types of fluents) that does not have parameters. We use the Feferman coding to add parameters to these events and fluents. However, in order to do so, we require extra tools in order to translate the coding of formulas into terms of the appropriate sort. We need to add some complete and decidable list of axioms in order to represent numerals in each of the domains of sort $\sigma^e, \sigma^f, \sigma^{\text{act}}, \sigma^F$. We will add a constant $\mathbf{0}^\sigma$ and the Presburger axioms of sort, for $\sigma \in \{\sigma^e, \sigma^f, \sigma^{\text{act}}, \sigma^F\}$.

We need also a list of axioms for the translation functions I^σ of sort $\sigma^{\mathbb{R}} \rightarrow \sigma$:

1. $I^\sigma(\mathbf{0}^{\sigma^{\mathbb{N}}}) = \mathbf{0}^\sigma$,
2. $I^\sigma \circ (\cdot)' = (\cdot)' \circ I^\sigma$. (where the $(\cdot)'$ in the left-hand side is of sort $\sigma^{\mathbb{N}}$ and the $(\cdot)'$ in the right-hand side is of sort σ .)

Finally, we need to add the axioms for the truth predicates. This list is the list presented in [19, p. 77].

This set of axioms is called EC.

Bibliography

- [1] George S. Boolos & Richard C. Jeffrey 1989, *Computability and Logic*. Cambridge University Press. Third Edition.
- [2] Michael Dummett 1976, What is a Theory of Meaning? (II). In Dummett, M. *The Seas of Language*. Oxford: 34-93. 1993.
- [3] Solomon Feferman 1984, *Toward Useful Type-Free Theories. I*. In *The Journal of Symbolic Logic*, Vol 49, No. 1: 75-111.
- [4] Gamut 1991, *Logic, Language and Meaning*. The University of Chicago Press. Vols 1,2.
- [5] Adele Goldberg 1995, *Constructions: A Construction Grammar Approach to Argument Structure*. The University of Chicago Press: Chicago and London.
- [6] Adele Goldberg 1997. Construction Grammar. In E.K. Brown & J.E. Miller (eds.), *Concise Encyclopedia of Syntactic Theories*. New York: Elsevier Science Limited.
- [7] Fritz Hamm, Hans Kamp, & Michiel van Lambalgen, *There is no Opposition Between Formal and Cognitive Semantics*. to appear.
- [8] Ray Jackendoff 1996, Semantics and Cognition. In Shalom Lappin (Ed.), *The Handbook of Contemporary Semantic Theory*, 539-559. Oxford, Blackwell.
- [9] Hans Kamp & Uwe Reyle 1993, *From Discourse to Logic, Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers.
- [10] Paul Kay 2002, *An Informal Sketch of a Formal Architecture for Construction Grammar*. In *Grammars* **5**: 1-19.
- [11] Ronald Langacker 1998, Conceptualization, Symbolization, and Grammar. In Tomasello, Michael (Ed.), *The New Psychology of Language: Cognitive and Functional Approaches to Language Structure*. Lawrence Erlbaum Associates: Mahwah, London.

- [12] David Lewis 1972, *General Semantics*. In Davidson and Harman (Eds.), *Semantics of Natural Language*. Reidel.
- [13] Laura A. Michaelis. forthcoming. *Construction Grammar*. In K. Brown (Ed.), *The Encyclopedia of Language and Linguistics*, Second Edition. Oxford: Elsevier.
- [14] Carl Pollard, 1996, *The Nature of Constraint-Based Grammar*. Pacific Asia Conference on Language, Information and Computation. Kyung Hee University. Seoul, Korea. Dec. 20. Http link: utkl.ff.cuni.cz/knihovna/pollard_talk1.pdf
- [15] Murray Shanahan 1999, *The Event Calculus Explained*. In *Lecture Notes in Computer Science*, Vol. 1600: 409-431.
- [16] Martin Stokhof & Jeroen Groenendijk 1991, *Dynamic Predicate Logic*. In *Linguistics and Philosophy*, Vol. 14, no. 1: 39-100.
- [17] Michael Tomasello 1998, Introduction: A Cognitive-Functional Perspective on Language Structure. In Tomasello, Michael (Ed.), *The New Psychology of Language: Cognitive and Functional Approaches to Language Structure*. Lawrence Erlbaum Associates: Mahwah, London.
- [18] Michiel van Lambalgen & Fritz Hamm 2003, *Event Calculus, Nominalisation and the Progressive*. In *Linguistic and Philosophy*, 26.
- [19] Michiel van Lambalgen & Fritz Hamm 2005, *The proper treatment of Events*. Blackwell Publishing.