Gregoromichelaki, E. (2011). “Conditionals in Dynamic Syntax”
In: Kempson, R., Gregoromichelaki, E. and Howes, C. (eds.).
The Dynamics of Lexical Interfaces. CSLI Publications.

8

Conditionals in Dynamic Syntax

Eleni Gregoromichelaki

8.1 Introduction

The purpose of this chapter* is to propose a general framework for the processing of expressions whose interpretation involves quantification over entities commonly referred to as *situations* or *eventualities* through an account based on the formal apparatus defined in the framework of *Dynamic Syntax* (DS, Kempson et al. 2001, Cann et al. 2005). As a case study we examine *conditionals*, a construction whose semantics involves a concept central to cognition and action but whose linguistic expression has baffled researchers for long time:

> Conditionals involve virtually every problem - logical or linguistic, descriptive or theoretical - that has ever been raised”(Smith & Smith 1988: 350).

Given this widely noted complexity (see also von Fintel to appear) it seems worthwhile to examine conditionals from the innovative analytic perspective suggested by DS, a formalism which approaches linguistic phenomena from the point of view of the processing mechanism.

Since parsing (and production) are arguably conducted incrementally in successive stages, their modelling necessitates the characterization of the partiality of information available at each stage and the formulation of time-linear constraints that define the alternatives open for further processing. These features of DS are taken to provide explana-

*I am very grateful to Ruth Kempson for invaluable comments and suggestions. Many thanks also to Ronnie Cann, Greg Mills and Lutz Marten. The work was supported by ESRC RES-062-23-0962 (DynDial).

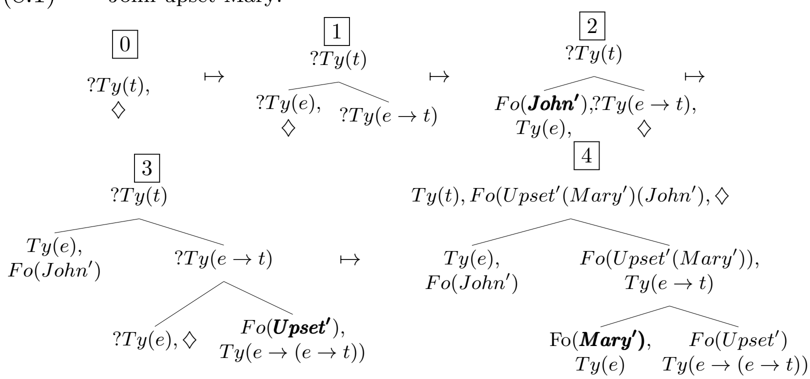
The Dynamics of Lexical Interfaces.
Ruth Kempson, Eleni Gregoromichelaki and Christine Howes (eds.).

tions of the syntactic properties of natural language (NL) expressions. In addition, the radical context dependence of NL processing suggests that the linguistic/grammatical specification of words or structures, which significantly under-specify their eventual interpretation, must include specified parameters that are resolved by reference to the general context. In the domain of conditionals where inherent vagueness and context dependence of interpretation has been noted by various researchers, the methodology then suggested by the DS perspective is to attempt to resolve the puzzles by defining unitary lexical/structural specifications which in combination with available processing strategies integrate both the contribution of the context of processing and the time-linearity of the available information at each stage. Therefore, our challenge here is to identify a suitably underspecified encoded content for *if*-clauses which can then be enriched either by input from its context of occurrence or in accordance with its time-linear presentation in order to provide the requisite interpretations, as well as, in parallel, explanations of the syntactic properties of such structures.

8.2 Dynamic Syntax

DS is a model of how interpretation is built up incrementally. Crucially, the output of any processing task is a representation of the content of a discourse uttered in a particular context (not a representation of some hierarchical structure defined over a string, i.e., not a *sentence type* abstracted out of various contexts). On this view, NL "syntax" is conceived of as the *process* by which semantic representations are dynamically constructed. Since pragmatics may interact at any point with the online interpretation process, output semantic representations (formulae) for the same string may differ as it is uttered in different contexts. Formally, DS is a lexicalized grammar using labelled word sequences and trees: the sequences are time-linearly presented strings of words with accompanying phonological, morphological and word-boundary specification; the trees formalize the semantic, functor-argument structure induced from utterances of such sequences in the form of (unordered) trees labeled by terms of a typed lambda calculus and other process-control labels. The basis of the formal characterisation is the (modal) logic of finite trees (LOFT: Blackburn and Meyer-Viol 1994) which permits addressing any node in the tree from the perspective of any other node using the *immediate-dominance* modalities $\langle\downarrow\rangle$ and $\langle\uparrow\rangle$ and variations over these. Such operators can be used to indicate nodes that exist already in the tree (e.g. $\langle\downarrow\rangle\alpha$ indicates that there is a daughter of the current node decorated by label α), with variants distinguishing $\downarrow_0/\downarrow_1$ as argument/functor daughter respectively, and Kleene * operations over these to define general *dominance* relations. The parsing process is driven relative to the imposition and subsequent satisfaction of requirements: $?X$ for any annotation X on a node constitutes a constraint on how the subsequent parsing steps must progress, i.e. X must be derived (eventually). Terms of only a small, fixed, set of semantic types are used (defined as the domain D_{Ty}), so no new types (functions) can be constructed. The main requirement is $?Ty(t)$, the type requirement to produce a formula of proposition type (t) from the word string; imposition of such requirements for other types, e.g., $?Ty(e)$, $?Ty(e \to t)$ impose constraints on tree development that formulae of the relevant types must be provided by the end of the parsing process. Such requirements may also be modal, e.g. while a decoration $?Ty(e)$ requires a term to be constructed at the current node, $?\langle\downarrow\rangle Ty(e)$ requires a daughter node to be so decorated. Consider in (8.1) a sketch of the general process of parsing as the induction of a sequence of partial trees, whose input is a one-node tree annotated with only the requirement $?Ty(t)$ and a pointer, $\diamondsuit$, indicating the node under development (the *Axiom*). The output is a binary branching tree whose nodes reflect the content of some propositional formula:[1]

(8.1) John upset Mary:



Processing proceeds by the execution of licensed *actions* that map one partial tree to another. These are defined in a language involving such commands as $\mathtt{make}(\langle\downarrow\rangle)$, $\mathtt{go}(\langle\downarrow\rangle)$, $\mathtt{put}(\alpha)$, $\mathtt{make}(\langle\downarrow_*\rangle)$, $\langle$IF ..., THEN ..., ELSE ...$\rangle$ etc. Sets of such actions incorporated in individual packages can be either general *computational rules* or *lexical actions* associated with words contributing content-formulae and other

[1] For purposes of illustration here we show the formulae specifications of nodes as simplified semantic terms (e.g. John' derived from the word *John*), i.e. we omit the details of the contribution of names and the lambda terms indicating predicates.

aspects of structure. *Computational rules* divide into two broad types: those for developing a tree-structure for which a number of strategies may be available (e.g. Introduction/Prediction); those annotating non-terminal nodes through algorithmic application of β-reduction and its associated type-deduction (Completion, Elimination). These in combination yield a resulting structure, with all nodes properly annotated and no requirements outstanding if a grammatical string has been processed.

Words are interpreted as instruction packages, including actions to construct parts of a lambda term (as values of the label Fo which takes values from a defined domain D_{Fo}) in a labelled tree representation. The packages are executed word-by-word from-left-to-right. The terminal nodes of the lambda term produced by a grammatical sentence do not necessarily stand in one-to-one correspondence with the words of the sentence as an individual word may induce sub-structure containing more than one labelled node; and there is no direct relation between the yield of the eventual tree and the sequential order of the string. For example, verbs in English are parsed when the pointer resides at a node decorated with $?Ty(e \rightarrow t)$ induced by a computational rule. The verb itself contributes not only a logical formula (e.g. $Fo(Upset')$) but also creates a new object node and moves the pointer to this node (the transition induced is that of transition 2-3 in Figure 8.1).[2] Parsing of an NP next will then provide the appropriate formula value for this node and rules like Anticipation, Completion and Elimination can then compositionally determine the combination of those formulae to satisfy the requirements remaining in a strictly bottom-up fashion.

The tree is incrementally constructed, but there are two ways to escape the strict linear processing order: by *structural underdetermination* and by *underdetermination of labels*; both may lead to delay in choices to be made. Structural underdetermination involves the addition of a sub-structure to the tree, an *unfixed node*, whose location in that tree is characterised as merely dominated by a previously constructed node ($\langle\uparrow_*\rangle Tn(\alpha)$) without, as yet, a fully specified hierarchical position. So, for example, in the processing of a string like *Mary, John likes* the *Adjunction rule licenses the introduction of a node into the tree to accommodate the content specified by the word *Mary* that cannot yet be given a fixed location in the tree skeleton under construction. Underspecification in the labels dimension is implemented by means of employing *meta-variables*, indicated as $Fo(\mathbf{U})$, $Fo(\mathbf{V})$,

[2]Formally: IF $?Ty(e \rightarrow t)$, THEN$\texttt{make}(\langle\downarrow_1\rangle), \texttt{go}(\langle\downarrow_1\rangle), \texttt{put}(Fo(Upset'), Ty(e \rightarrow (e \rightarrow t))), go\langle\uparrow_1\rangle, \texttt{make}(\langle\downarrow_0\rangle), \texttt{go}(\langle\downarrow_0\rangle), \texttt{put}(?Ty(e))$, ELSE Abort.

etc., as temporary labels requiring their substitution by terms from context (given constraints such as gender, person, and so on). Thus pronouns and other anaphoric elements (e.g. ellipsis sites) provide such metavariables as initial formula values which are required (through the injunction $?\exists\mathbf{x}.Fo(\mathbf{x})$) to be instantiated (through the rule of Substitution) at some point during the parsing process.

8.2.1 Quantification and anaphora

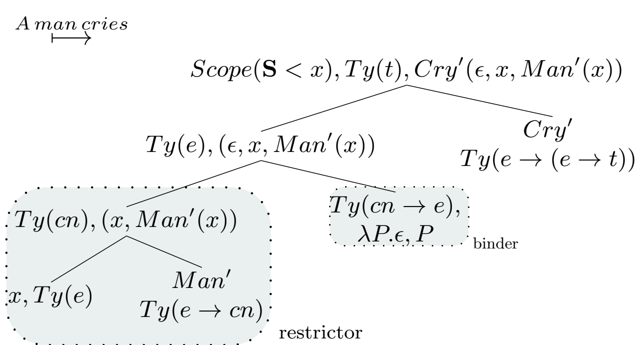
All noun phrases, even quantified ones, project terms of type e. Quantifying expressions are analysed in the manner of arbitrary names of predicate-logic natural deduction, as formulated in the *epsilon calculus* of Hilbert and Bernays (1939), a conservative, but more expressive, extension of predicate logic. Introduction of such terms in the language is based on the following equivalence:

$$\exists x.F(x) \equiv F(\epsilon x.F(x))$$

This indicates that an existential statement is equivalent to one in which a *witness* of the truth of the statement can appear as an argument of the statement's predicate. Such a witness appears as an *epsilon term* containing as its *restrictor* the predicate itself and denotes some arbitrary entity satisfying the predicate (if such an entity exists, if it doesn't any arbitrary entity). The dual of an epsilon term is a tau-term ($\tau - term$) which is equivalent to universal quantification.

Exploiting these equivalences, in DS, such terms are compositionally constructed through the time-linear processing of determiners, which introduce the binders (below $Ty(cn \rightarrow e), \lambda P.\epsilon, P$) and common nouns which contribute the restrictor (below $Ty(cn), (x, Man'(x))$). Both combine to produce complex subtrees of $Ty(e)$ annotated with an epsilon calculus term becoming eventually the argument of the predicates contributed by verbs and VPs:

(8.2)



Quantifier scope is not expressed as part of the tree architecture but through scope constraints collected incrementally during the parse process. Scope dependencies ($<$) are declared as collected statements stored in the label *Scope*(...).[3] Epsilon calculus terms are employed by DS because they may carry a record of the context within which they occur inside their restrictor. According to defined computational rules of Quantifier Evaluation (see Kempson et al. 2001), an intermediate interpretation of a sentence such as *A man cries* will take the form $Fo(Cry'(\epsilon, x, Man'(x)))$ derived by simple functional application as shown in (8.3). But this will be eventually algorithmically transformed to a formula where an appropriate epsilon term, abbreviated as a below, appears as the argument of the conjunction of the predicates contributed by the common noun and verb:

(8.3) $\overset{A\ man\ cries}{\longmapsto} Cry'(\epsilon, x, Man'(x)) \overset{\text{Q-Evaluation}}{\longmapsto}$
$Man'(\alpha) \wedge Cry'(\alpha)$ where $\alpha = (\epsilon, x, Man'(x) \wedge Cry'(x))$

On the other hand, a $tau - term$ will induce universal quantification:

(8.4) $\overset{Every\ man\ cries}{\longmapsto} Cry'(\tau, x, Man'(x)) \overset{\text{Q-Evaluation}}{\longmapsto}$
$Man'(a) \rightarrow Cry'(\alpha)$ where $\alpha = (\tau, x, Man'(x) \rightarrow Cry'(x))$

Multiple quantification of course yields more complex restrictor specifications, but the evaluation algorithm applies to arbitrarily complex combinations of terms and with variation in the connective depending on the type of quantifier involved. The restrictor of all such terms contains a record of the propositional structure that gave rise to it so that it provides a suitable antecedent for subsequent cases of *E-type anaphora* (see Kempson et al. 2001).The effect of the interpretation of pronouns as bound variables is achieved via the independent assumptions of the content of pronouns as providing invariably metavariables and the Quantifier Evaluation rules. For example, processing the string in (8.5) below will initially produce a representation where a metavariable **U** will temporarily occupy the argument position. Subsequent substitution of this metavariable with the unevaluated tau-term in subject position (see (8.7)) will eventually lead to a logical form where both subject positions of the two propositions involved are occupied by variables bound by the tau binder:

(8.5) Every student$_i$ believes he$_i$ is clever.

(8.6) $Believe'(\tau, x, Student'x)(Clever'(\mathbf{U}))$

(8.7) $\overset{\text{Substitution}}{\longmapsto} Believe'(\tau, x, Student'x)(Clever'(\tau, x, Student'x))$

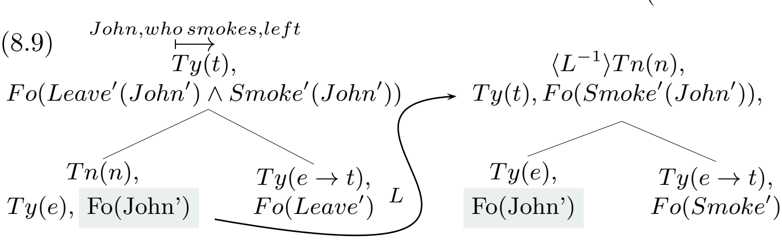
[3]**S** in (8.2) above is a metavariable that can be substituted by the index of evaluation for the proposition.

(8.8) $\stackrel{\text{Q-Eval}}{\longmapsto}$ $Student(\alpha) \rightarrow [Believe'(\alpha)(Clever'(\alpha))]$ where
$\alpha = \tau, x, [Student'(x) \rightarrow Believe'(x, (Clever'(x)))]$

The effect that results from these various forms of underspecification and update is a process of dynamic transitions through partial trees and annotations that leads to a licensed eventual logical form representing the proposition expressed in a particular context. In DS, all syntactic phenomena are reformulated in these terms, leading to the comprehensive claim that the NL syntax does not involve a separate level of representation besides what is needed for semantics (Cann et al 2005, and elsewhere).

8.2.2 Subordination in DS

Conditionals present one of a number of cases where there is tension between analyses in terms of the traditional notions of *coordination* and *subordination.* As regards our purposes, DS distinguishes only two ways of combining the input provided by the processing of distinct clauses in order to produce a complex proposition as their content:
(a) a clause can contribute the argument of a predicate as in, e.g., the propositional object of a verb; in this case the *local tree* will consist of two predicate-argument structures one embedded into the other, or,
(b) two clauses can introduce two separate trees connected by a LINK relation between their nodes. This mode of processing results in an overall structure called the *global tree* which can encompass any number of LINKed trees and is a representation equivalent to an inferential/cognitive unit (cf. Blakemore 1987, Carston 2002). LINKed trees are associated through the LINK modality, $\langle L \rangle$. Adjuncts, coordinations and other structures are processed by employing this device which allows incorporation within a tree of information that is to be structurally developed externally to it. Relative clauses, being one core case of adjunction, are processed through the construction of a LINKed tree bearing a requirement that it contains as a sub-term the formula on the source-node from which the LINK relation is defined ($John'$ below):

(8.9)


In the building up of such paired LINKed trees for relative clauses, the requirement imposed for a common term is satisfied by the processing of the relative pronoun which induces the copying of that term. As this copy is initially introduced on an unfixed node, it is then necessarily constrained to appear within the newly emergent propositional structure (reflecting the *island constraints* associated with such structures).

8.3 Conditionals: structure and processing

8.3.1 Syntactic properties: the tension between coordination and subordination

The analysis of an NL conditional structure under the truth-functional material implication interpretation and a compositional view of the syntax-semantics interface implies a logical form where the two clauses are conjoined by a two place connective and are equivalent in status. Under one such possible account, NL *if . . . then* can be viewed as a discontinuous connective relating two sentences of equal status (see e.g. Chomsky 1957: 22, Strawson 1986). However, even disregarding the fact that the appearance of *then* is not necessary in conditionals, the relation between antecedent and consequent is unlike that of two conjuncts related by *and/or/but* (see e.g. Geis 1985). Instead, *if*-clauses seem to pattern more with another type of structure, also analysed in terms of the LINK relation, namely, *relative clauses*. First of all, *Right Node Raising* is precluded between antecedent and consequent as is also the case with relative clauses:

(8.10) John will support and Mary will try to promote the manager of their department

(8.11) *If John supports ___ Mary will try to promote the manager of their department

(8.12) *[Mary will support ___] [if John tries to promote ___] the manager of their department

(8.13) Mary will support John Petropapadopoulos, who Bill met yesterday ⇒

(8.14) *[Mary will support ___] [who Bill saw ___] John Petropapadopoulos.

Two clauses related by a conjunction and can be made to share a common subject (Conjunction Reduction). This structure is not allowed for *if*-clauses (and relatives alike):

(8.15) John [bought the newspaper] and [sold the milk]

(8.16) *John [bought the newspaper] if [sold the milk]

(8.17) *If John [sold the milk] [bought the newspaper]

(8.18) *Mary [will support John], who [met yesterday]

Another difference that points to distinct analyses for conjunctions on the one hand and *if*-clauses (and relatives) on the other is the phenomenon of *Gapping*:

(8.19) John will buy the newspaper and [Mary ___ the milk]

There is no corresponding construction for *if*-clauses (neither with relatives):

(8.20) *John will buy the newspaper if Mary ___ the milk

(8.21) *John saw Bill who ___ Mary

Coordinated clauses allow for the phenomenon of *Across-the-Board extraction*:

(8.22) The man who [John dislikes ___] and [Mary loves ___] came to see me yesterday

(8.23) Who [did John dislike ___] and [Mary love ___]?

If-clauses do not generally permit extraction:

(8.24) Who did John dislike ___ if Mary saw ___ ?

The fact that a conditional consists of two asymmetrically related clauses is also supported by the fact that the *if*-clause cannot bear independent speech act indicators, that is, the verb form inside the *if*-clause cannot be in Imperative or Interrogative mood:

(8.25) *If come to the partly it will be fun

(8.26) *It will be fun if come to the party

(8.27) *If are you busy you will come to the party?

In this respect *if*-clauses resemble subordinated clauses and relatives which are also unable to have independent speech act indicators:

(8.28) *John asked that did Mary come home?

(8.29) *John said that come home

(8.30) *John, who did you like, came home

Since there are conditional assertions, questions and commands, these are indicated by the grammatical form of the consequent which is what has motivated the traditional claim that the consequent provides the primary structure and the *if*-clause is an adjunct:

(8.31) If you are not busy come to the party.

(8.32) If you are not busy are you coming to the party?

(8.33) Will you come to the party if you are busy?

The *if*-clause, like other adjuncts, can appear in different positions without obvious truth-conditional effects:

(8.34) If John shouts, Mary gets upset.

(8.35) Mary gets upset if John shouts.

A syntactic account of *if*-clauses then needs to provide the resources for processing them at different positions. Moreover, there is no strict adjacency requirement between antecedent and consequent. The consequent corresponding to a clause-initial antecedent might be embedded as can be seen in the cases below:

(8.36) [If John leaves] Mary believes that Bill will stay =
a. Mary believes that if John leaves Bill will stay
or
b. If John leaves then Mary believes that Bill will stay.

(8.37) If the dinner had been ready Mary believes that John would not have complained
a. [If the dinner had been ready] (then/in that case) Mary believes that John wouldn't have complained
or
b. Mary believes that [if the dinner had been ready] (then/in that case) John would not have complained

However, the distance between antecedent and consequent is not arbitrary. Their separation respects what in the linguistic literature are sometimes called *island restrictions*, e.g., the *Complex NP-constraint*, Ross 1967):

(8.38) Mary called the man [who will be hired if John leaves]. ⇒
*If John leaves Mary called the man [who will be hired].

This seems to imply that antecedent and consequent stand in some kind of local relation that has to be expressed. Moreover, extraction is not allowed from inside an *if*-clause, i.e., *if*-clauses, like relative clauses, are themselves islands:

(8.39) John saw the man [who shot Mary] ⇒

(8.40) *Who did John see the man [who shot ___]?

(8.41) John will fire Mary [if she calls Bill]. ⇒

(8.42) *Who will John fire Mary [if she calls ___]?

(8.43) If John sees Mary he will kiss her. ⇒

(8.44) *Who [if John sees ___] will he kiss her?

These syntactic facts point then to the conclusion that *if*-clauses behave as some kind of relative clause. However, in the present context, given that DS takes a processing perspective, such syntactic evidence can be taken in two ways: either as indications of the structural properties of the eventual representation derived or as properties of the processing apparatus utilised to derive this final representation. Unlike other frameworks where syntactic issues are very generally taken as independent of and prior to semantics, in DS, NL "syntax" is taken to be nothing more than the progressive compositional construction of some logical representation. This logical representation has its own syntax as a domain of LINKed trees. But, as we said earlier, such trees neither reflect word order nor can be associated one-to-one with single strings since there can be various parsing routes deriving the same tree structure, a structure which directly reflects the interpretation assigned to the NL input processed. So now we need to examine what kind of interpretational facts we need to take into account in the processing of conditionals and how these relate to the syntactic restrictions mentioned above.

8.3.2 Interpretational effects in conditionals

Haegeman (2001, 2003) claims that, syntactically, there appear to be two entirely different types of conditionals:[4]

(a) *event conditionals* in which the *if*-clause modifies the event denoted by the main clause:

(8.45) If it rains we will all get terribly wet and miserable

(b) *premise conditionals* in which the *if*-clause contributes a proposition which has to be taken as "the privileged context" for the processing of the consequent:

(8.46) If (as you say) it is going to rain, why don't we just stay at home and watch a video?

In terms of semantics, Haegeman argues that the antecedent in an event conditional provides a "cause" leading to the "effect" denoted by the content of the main clause. Premise conditionals, on the other hand, are usually "echoic" and have their own illocutionary force. So, in terms of structure, event conditionals are fully integrated in the (syntactically encoded) speech act of the main clause so that the semantic differences between the two types of conditional have a transparent syntactic basis. *If*-clauses in event conditionals are merged inside the VP/IP domain whereas premise-protases are located in the CP domain (therefore the two types exhibit distinct *external syntax*). Haegeman also assigns distinct internal syntactic articulation to each type of *if*-clause (their *internal syntax*). Event conditionals, according to Haegeman, are reduced clauses in that they lack the full CP articulation, e.g. (illocutionary)

[4] Iatridou (1991) distinguishes three types of conditionals: *hypothetical*, *factual* and *relevance* conditionals. For factual conditionals we assume that an analysis as in Noh (1998) will assimilate them under a unified type with hypothetical ones.

force (and other) heads are missing, while premise conditionals are intact in that they include the same CP structure as main clauses. Shaer (2003) and Gregoromichelaki (2006) provide data showing that there are no definitive criteria that distinguish the two types of conditional that Haegeman postulates, for example, "temporal subordination" and "root transformations" are equally possible in both types (and, in any case, the existence of a strict two-way ambiguity in terms of semantics is also in doubt as Edgington 2003 argues). Hence, since both types of conditionals seem to be uniform in terms of syntactic properties a unified account should be pursued.

However, a widely adopted unified account of conditionals also faces problems: examining cross-linguistic distributional evidence Haiman (1978, 1993) has claimed that *if*-clauses are generally *topics*. From the present perspective, this claim is contradicted by the fact that *if*-clauses can occur post-verbally (and this without any necessary indication of a revision or afterthought construal). Notice especially that such postverbal *if*-clauses can accommodate anaphoric elements which depend for their content on the previous clause and contribute to the complex predicate attributed to the subject:

(8.47) Every student$_i$ will succeed if he$_i$ is not lazy.

If *if*-clauses were consistently analysed as "topics", i.e. introducing background or given information, external to the main assertion, such dependencies are not easily explained. Moreover, although it has been claimed (Rooth 1985) that *if*-clauses, unlike *when*-clauses, cannot be focussed, von Fintel (1994) and Bhatt (1996) show that they can both be the new information conveyed by an answer to a question and bear nuclear accent:

(8.48) A: What would motivate John to shave?
B: John always shaves, [if his MOTHER is coming to visit]$_F$

So an analysis that takes *if*-clauses to uniformly introduce "topics", somehow loosely associated with the main clause, can reflect neither their intepretational nor their informational structure and syntactic properties. On the other hand, a syntactic ambiguity account, as in Haegemann (2003), is also undesirable since, at least in English, there is no justification for positing two distinct underlying structures.

8.4 A DS view of conditionals

The present account is an attempt to make sense of these very mixed data while adopting what is a broadly agreed set of semantic assumptions about conditionals. The parallel properties of *if*-clauses and nominal relatives, which contrast with the properties of coordinate structures, have led several researchers to the assumption, which will be adopted here, that, in some sense to be made precise, *if*-clauses fall within the same class of structures as relative clauses in the same way that *when*-clauses and *where*-clauses can be taken as relatives over implicit time or location specifications (see e.g. Geis 1985, Bhatt & Pancheva 2001). What is then crucial for us here is to determine what kind of element an *if*-clause attaches to as a relative which brings us to the issue of what type of semantic representation we should assume to be constructed out of the processing of a conditional sentence. Since, in DS, syntactic and semantic issues are fundamentally inter-dependent this means that the semantic representation and its derivation as the outcome of parsing a conditional should crucially be compatible with the syntactic facts mentioned above.

From a structural point of view, in DS terms, the parallels in the behaviour of *if*-clauses and relatives point to the conclusion that an analysis in terms of LINKed structures should be assigned to both which will explain the data in (8.40)-(8.44) regarding their status as islands with respect to extraction. However, the kinds of movement allowed for the *if*-clause shown above in (8.34)-(8.37) indicate that there is a difference: relative clauses cannot appear freely in any position unless they follow a dislocated argument to which they attach (besides cases of extraposition). This shows that *if*-clauses behave more like arguments in terms of their positioning within the sentence. In addition, in the same way that arguments cannot be dislocated out of "islands", the same restriction holds for *if*-clauses as shown in (8.38). This means that in order to capture the parallels between *if*-clauses, arguments and relatives there has to be an argument node inside the tree representation constructed from the consequent to which the content of the *if*-clause attaches via the LINK relation. The licensed variability in the positioning of arguments would then allow us to account for the freedom of movement of the *if*-clause.

In terms of semantics, along with other researchers, we will assume here that *if*-clauses are relatives that introduce the restrictor of a term inducing universal quantification over a domain of contextually specified events or situations (see e.g. Lycan 2001). This term is a $Ty(e)$ argument of the main clause, the consequent. Even though the data shown in (8.47)-(8.48) are problematic for the *conditionals-as-topics* thesis they are not problematic for the analysis of *if*-clauses as contributing restrictors to quantificational terms. As von Fintel (1994) shows, restrictors in nominal terms can also be focused:

(8.49) A: Who here is clever enough to solve this problem?
B: Well, most of the [GRAD STUDENTS]$_F$ should be able to help you.

For the present analysis of *if*-clauses as relatives we can observe that elements in a relative clause can also be focused:

(8.50) A: Did you see the man that John dislikes?
B: No, but I saw the woman that John [ADORES]$_F$

(8.51) A: Who did John hit?
B: John hit whoever [APPROACHED]$_F$

We take these as evidence that *if*-clauses provide a consistent input as far as their contribution to the semantic representation is concerned, i.e. they provide specifications of the $Ty(e)$ situation argument of the representation. Under the DS assumptions made here, an appropriate pragmatic theory should be more suitable to derive the exact interpretation associated with the processing of conditional structures in context without the semantic representation explicitly encoding every type of effect (which might very well be an open-ended number cf. Declerck and Reed 2001). But, in general, it is widely accepted that a sentence-initial *if*-clause somehow provides the context for the processing of the consequent. We can give formal substance to this intuition in an illuminative way within DS. Without attributing any ambiguity to the encoded content of the *if*-clause itself (unlike Haegeman 2003), exploiting the same mechanisms available in the processing of NPs, we can assume that *if*-clauses too can be processed either as LINKed structures occupying independent subtrees or as providing content occupying structurally underspecified (unfixed) nodes. We should then expect that, as is the case with free word order languages and NP-dislocation phenomena, distinct parsing routes will be exploited for contextual effects facilitating one type of interpretation over another (see, e.g. the distinction between "focus" and "topic" NP-interpretations, Cann et al. 2005).[5] So, in some cases, *if*-clauses can convey new/non-backgrounded information and we assume that the analysis as annotating unfixed nodes is more appropriate:

(8.52) A: Under what conditions are you prepared to surrender?
B: [(Only) If JOHN surrenders]$_F$ I might do.

(8.53) A: Are you going to play soccer on Sunday?
B: We'll play [if the SUN shines]$_F$ (von Fintel 1994: 82)

[5] Although evidence for such distinct parsing strategies for *if*-clauses is not readily available in English (apart from intonation), in languages with V2 we find two syntactically distinguished ways of associating the *if*-clause with the consequent.

On the other hand, we assume that the specific interpretation usually characterised in the literature as *topic*, i.e. "given" or "background" information, is only available at the left-periphery under a construal in which a left-peripheral *if*-clause is processed as the *head*, the point from which the LINK relation originates. The situation argument that is introduced with this means will be then necessarily unified with the situation argument of the main clause. This is a natural characterisation since only in this structure can we assume that what is provided initially is introduced explicitly as the background for processing the following assertion. For cases of LINKed subtrees contributed by right-peripheral *if*-clauses we will see that they provide confirmation for the choice of an already contextually given element. As these effects are exactly parallel to those widely observed in the processing of nominal phrases the approach taken here is in line with Schlenker's (2006) view regarding the fundamental symmetry that underlies linguistic reference in ontological domains like individuals, times and worlds despite the apparently variable syntactic means employed in individual languages.

We will now proceed to show how these assumptions can be implemented with minimal extensions to standard DS apparatus. First we will show how to incorporate the situation argument as occupying its own node in the standard DS tree representations (8.4.1) and then how the processing of *if*-clauses targets this particular node (8.4.2). We will then see two additional ways of processing *if*-clauses, as unfixed nodes (8.4.4) or LINKed -trees (8.4.4), exactly as any other argument and with similar interpretational effects.

8.4.1 Introduction of the situation argument

Following much current work in the formal semantics literature, we will employ an additional argument for propositional representations standing for the situation of evaluation (see Heim 1990, von Fintel 1994, Chierchia 1995 a.o.). Farkas (1997) proposes that for each world w we define an extensional model $\mathcal{M}_w =< S_w, U_w, V_w >$ where S_w is a set of situations in w, U_w is the set of individuals in w and V_w assigns values to the constants of the language with respect to the situations in S_w. As suggested in Kratzer (1986), we assume that situations are parts of worlds, each situation part of a unique world. Worlds are then defined as maximal situations. Truth of a logical form (*lf*) in a world w is determined with respect to truth in a situation in w:

(8.54) An *lf* is true in w with respect to $\mathcal{M}$ iff there is a situation s in S_w such that the *lf* is true in s with respect to $\mathcal{M}_w$.

Adapting this assumption to DS terms, the situation argument of a predicate will be explicitly represented on the tree and will combine with it by the usual means of function application. So we will add a new type in D_{Ty} in order to allow the situation argument to be processed. We will call this new type $Ty(e_s)$. We assume then that the values in D_{Ty} are sorted with $Ty(e)$ as a general type with subtypes of $Ty(e_s)$ for situations and $Ty(e_i)$ for individuals, $Ty(e_w)$ for worlds etc. However, for simplicity of illustration we will continue to notate the type of individuals as $Ty(e)$, i.e. we will omit the subscript when no ambiguity arises. Metavariables can be specified to take values either of the most general type ($Ty(e)$) or the more specific types ($Ty(e_i)$, $Ty(e_s)$ etc.). The content assigned to verbs (and predicates in general but we omit discussion of this issue here) comes from the lexicon with an additional situation argument which combines with a situation term available from the context.

In order for the situation argument to be introduced through the usual DS processing apparatus,[6] the rules of Introduction and Prediction will be employed to provide for a further $?Ty(e)$ position. Introduction and Prediction starting from the $?Ty(t)$ axiom:

(8.55) $?Ty(t), World(w_0), Scope(w_0), \diamondsuit$

induce two additional nodes: a node with a requirement for $Ty(e_s)$ and its sister $?Ty(e_s \rightarrow t)$. The scope statement ($Scope(...)$) includes the world of evaluation which we have assumed to be w_0, the actual world as a default (see Papafragou 1996: 186-187, 2000 for justification in the context of a pragmatic theory). The situation argument can be initially introduced as a metavariable, $\mathbf{S_i}$, $\mathbf{S_j}$, $\mathbf{S_k}$ etc., since it is possible to supply a value for it from the context (as a instance of *saturation*, see e.g. Recanati 1999). In the tree below, with the pointer at the type e_s node the metavariable will be inserted by a computational rule.[7] A requirement for the term substituting for the metavariable to participate in some statement is also inserted:[8]

[6]Gregoromichelaki 2006 presents an alternative way of introducing the situation argument as an optional addition to the representation in case it is needed (as e.g. in the case of a conditional structure). This option follows Recanati (1999) and is the one compatible with Cann (this volume) account of tense. As we are not dealing with tense and modality here we follow the simpler route.

[7]Note that Anticipation can move the pointer downwards if the situation argument is to be developed immediately afterwards.

[8]For exact specification of the rules see Gregoromichelaki 2006, Ch 4.

(8.56) $\text{INTRODUCTION} \longmapsto \text{PREDICTION} \quad \textit{Situation Metavariable Introduction} \longmapsto$

$$?Ty(t), Scope(w_0 < \mathbf{S}),\ ?\exists\mathbf{x}.Scope(w_0 < \mathbf{x})$$

$$Ty(e_s), Fo(\mathbf{S}), \Diamond \qquad\qquad ?Ty(e_s \rightarrow t)$$

If there is no appropriate value available for the metavariable the underspecified value and requirements can be left to await resolution until the latter stages of the parse. Any quantificational terms introduced subsequently will also be able to depend on this metavariable. Crucially for the present analysis, the situation argument can be a variable, an epsilon term or a tau term. These terms will contribute to the scope statement as any other regular argument. This is motivated by the interaction between quantification over individuals and situations. Simplifying somewhat, the sentence below can be taken as ambiguous between the two logical forms displayed below it:

(8.57) Henry gracefully ate all the crisps

(8.58) ∀y. Crisp'(y) → ∃e. Eat'(Henry', y, e) ∧ Graceful'(e)

(8.59) ∃e. ∀y. Crisp'(y) → Eat'(Henry', y, e) ∧ Graceful'(e)
from Taylor (1985)

In addition, Farkas (1997) argues that the situation argument must scopally interact with individual quantificational terms in order to derive the range of interpretations possible for the following:

(8.60) If *a boy he likes* comes over, Johnny shows *him* his turtle.

The indefinite in the above can be interpreted as having either wide scope with respect to the situations of evaluation, in which case there is a particular boy being mentioned, or it can have narrow scope in which case for each situation considered there is potentially a different boy involved. The obvious way to deal with these cases in DS is to allow terms representing situations to appear in the scope statement and interact freely with the variables contributed by the nominal terms. As is standard in DS, we assume that indefinites contribute epsilon terms which must necessarily depend on some other term in the current tree. In the case above in order to derive the wide scope of the epsilon term derived from the indefinite, we need a DS representation in which this term outscopes another term representing the range of situations introduced by the conditional. The term derived from the indefinite will in turn be outscoped by the world of evaluation which must be taken as necessarily the first element in the scope statement.[9]

[9] Although we will not deal with modality and tense issues in the present work we will assume that the world of evaluation w_0 will also eventually appear as a

The rest of the parsing rules remain as usual. As regards the tree in (8.56), the $?Ty(e_s \rightarrow t)$ node (which is equivalent to the $?Ty(t)$ node in the standard presentations of DS) can now be expanded with Introduction and Prediction to accommodate the subject and the predicate. We can now process the lexical input provided by the subject, e.g. *John*, and this will decorate the type e node. Pointer movement will proceed to the predicate node where the verb can be parsed. The modified lexical entry for an intransitive verb like *run* is shown below (tense and mood specifications are omitted):

(8.61) *run*

IF	$?Ty(e \rightarrow (e_s \rightarrow t)), \Diamond$
THEN	$\mathtt{put}((Fo(\lambda x.\lambda s.Run'(x)(s)), Ty(e \rightarrow (e_s \rightarrow t))))$
ELSE	$\mathtt{abort}$

An epsilon term available from the context can now be provided to resolve the metavariable at the $Ty(e_s)$ node, we annotate this in schematic form as $\epsilon, s, p(s)$. By means of the Q-Evaluation Rule the formula on the root node will be converted to a quantificational structure where it is declared that there is a situation that satisfies the descriptions p and $Run'(John')$:

(8.62) $\xrightarrow{\text{...Completion... Q-Evaluation}}$

$$Ty(t), \Diamond, Scope(w_0 < s), Fo(Run'(John')(\epsilon, s, p(s))) \overset{Q-Eval}{\longmapsto}$$
$$Fo(w_0 : p(\alpha) \wedge Run'(John')(\alpha) \text{ where } \alpha = \epsilon, s, [p(s) \wedge Run'(John')(s)])$$

$Ty(e_s)$, $Fo(\epsilon, s, p(s))$

$Ty(e_s \rightarrow t)$, $Fo(\lambda s.Run'(John')(s))$

$Ty(e), Fo(John')$

$Ty(e \rightarrow (e_s \rightarrow t))$, $Fo(\lambda x.\lambda s.Run'(x)(s))$

According to the semantics presented in (8.54) above, the *lf* that is the Fo value at root node of the tree in (8.62):

(8.63) $Fo(w_0 : p(\alpha) \wedge Run'(John')(\alpha)$ where
$\alpha = \epsilon, s, [p(s) \wedge Run'(John')(s)])$

---

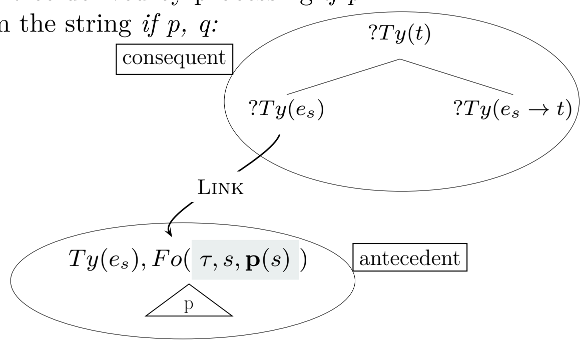
label on the formula decorating the root node of a propositional tree as is standard in DS. This is because we do not expect that there are any scope ambiguities regarding terms and the world of evaluation for each proposition. Being the label, the world of evaluation has necessarily widest scope and acts as a point of reference for the evaluation of all other terms. The actual world w_0 is the default world of evaluation. Modals and other intensional operators can be seen to optionally introduce new worlds of evaluation by existentially or universally quantifying over worlds accessible from w_0.

is true in world w_0 iff there is a situation s satisfying the restrictor p in w_0 that also satisfies the nuclear scope, $Run'(John')$, in w_0. We now have most of what is needed to allow the processing of conditional sentences. Let's see what more needs to be added.

8.4.2 Conditionals: logical form

Under the present analysis sentences containing *if*-clauses will be assumed to give rise to two LINKed trees: processing of the main clause, the *consequent*, results in one tree, the *main tree* from now on, while processing of the *antecedent* results in another. The latter appears connected to the situation node of the main tree by the LINK relation. The establishment of this relation will also effect a unification of values between two nodes of the joined trees in the following way: the final result of processing the antecedent of a conditional will produce a tau term ($\tau - term$) as the root of the LINKed tree (highlighted in (8.64)):[10]

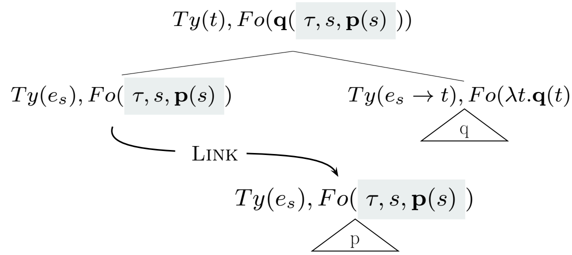
(8.64) The tree derived by processing *if p* from the string *if p, q:*



This term once derived will be copied in the main tree in order to serve as the Formula value of its situation node. The newly introduced τ-term in the main tree, being an ordinary argument of the predicate, will be incorporated in the proposition derived at the root through the usual function-application process:

[10] We omit reference to the world of evaluation from now on. We also omit scope statements and any other irrelevant detail.

(8.65) The *global tree* derived from *if p, q:*



Since the shared term is a quantificational τ-term, it will contribute to the Scope Statement of the main tree. The DS rules for processing quantification structures (Q-Evaluation) will apply at the end of the parse and yield a propositional structure at the root node:[11]

(8.66) Initial formula value derived by parsing *if p, q*:
$q(\tau, s, ps)$
Formula that results after application of Q-Evaluation Rule:
$p(\alpha) \rightarrow q(\alpha)$
where $\alpha = \tau, s, (ps \rightarrow qs)$

We thus derive a proposition where the content of the consequent (q, above) is construed as a property of situations. What is asserted is that this property q is true of all the situations that satisfy the description p provided by the content of the *if*-clause: $q(\tau, s, ps)$. That is, the content of the *if*-clause, p, becomes the Restrictor in a universal quantification over situations (τ, s, ps) whereas q becomes the Nuclear Scope. According to the semantics proposed in (8.54) above, the *lf* in (8.66) is true in a world w iff all the values of s that satisfy the restrictor, p, in w also satisfy the nuclear scope, q, in w. By the Q-Evaluation rules we also derive a term (α in (8.66) above) which can serve as the situation argument for another proposition or as the replacement for the metavariable contributed by a pronominal.

Since we want to capture the contextual dependency of situational quantification[12] as an instance of *saturation* (see Recanati 1999) we will assume that the Restrictor itself includes a metavariable that has to be substituted by appealing to the context of utterance. This means that only situations that are considered "relevant" will be included in the set of situations quantified over according to the Restrictor. Although we will not include it in the tree representations for simplicity,

[11] We omit brackets freely for readability.

[12] This is a standard assumption, see e.g. Lycan's 2001 "envisaged" and "real" possibilities.

we assume therefore that the Restrictor includes a metavariable that has to be substituted by the context of utterance: $\tau, s, (ps \wedge \mathbf{R}s)$. $\mathbf{R}$ is a metavariable that has to be substituted by a predicate of situations (a complex proposition) expressing "what is expected" or "envisaged" given the propositions in the *Discourse Context* (see also von Fintel 1994).[13] This will allow us to derive the distinct flavours associated with the meaning of types of conditionals that have been discussed in the literature without having to make syntactic distinctions encoding those differentiations. Notice also that, according to DS, given that the grammar interfaces with context incrementally at each point, propositions derived by processing NL sentences can also incorporate inferentially derived content, for example, not just the bare proposition expressed but also any higher level explicatures as argued in Purver et al (2010). This allows us to address some problematic aspects of conditionals. For example, Noh (1998) analyses "metarepresentational" conditionals which show a pattern problematic for standard truth-functional accounts:

(8.67) A: Two and eleven makes thirty
B: If two and eleven makes thirty, you need more work on maths. Noh (from Noh(ibid))

As Noh argues, such cases (and many others) can be accommodated if we assume that the content assigned to the antecedent is (meta)representing not simply the proposition expressed by the previous utterance but rather the higher-level explicature associated with it. So the proposition expressed by the antecedent is not simply "two and eleven makes thirty" but rather "if *you say/believe that* two and eleven makes thirty". As in DS no representation over strings is defined, the (optional) derivation of speech act information will naturally license this interpretation that accommodates the content of this higher level explicature plus any other contextually relevant assumptions. Under this view, the final propositional content expressed by B's utterance in (8.67) above will be: "All situations in which you believe/say that two and eleven makes thirty . . . are situations in which you need more work on maths".

In more concrete terms, let's look at the completed tree derived for a sentence like *if John cries, Mary laughs.* The diagram is shown below:

[13] Note that the *Discourse Context* is not assumed to contain only information that has been made available by processing linguistic input; it can be extended by means of inference.

(8.68) $\textit{if John cries} \overset{}{\longmapsto} \textit{Mary laughs}$

$Ty(t), Fo(Laugh'(Mary')(\tau, s, Cry'(John')(s)))$

$Ty(e_s),$ $Fo(\tau, s, Cry'(John')(s))$ $\qquad$ $Ty(e_s \rightarrow t),$ $Fo(\lambda t.Laugh'(Mary')(t))$

Link

$Ty(e_s),$ $Fo(\tau, s, Cry'(John')(s))$

$Ty(cn_s),$ $Fo(s, Cry'(John')(s))$ $\qquad$ $Ty(cn_s \rightarrow e_s),$ $Fo(\lambda P.\tau, P)$

$Ty(t),$ $Fo(Cry'(John')(s))$ $\qquad$ $Ty(t \rightarrow cn_s),$ $Fo(\lambda R.s, R)$

The $Ty(t)$ proposition derived by the processing of the antecedent, $Cry'(John')(s)$, becomes the argument of a function that copies its situation argument (which must be a free variable, s in (8.68) above) and creates a Restrictor in the appropriate format required for the construction of a τ-term: $s, Cry'(John')(s)$. The τ-binder is then applied to the Restrictor by means of function application and the τ-term is eventually derived: $\tau, s, Cry'(John')(s)$. This term is subsequently copied at the situation node of the main tree and serves as the situation argument there. The proposition decorating the root node of the global tree, $Laugh'(Mary', (\tau, s, Cry'(John')(s)))$, can be interpreted as:

"Mary laughs in all situations that John cries"

or

"All situations in which John cries are situations in which Mary laughs".

Hence, the role of the content of the *if*-clause here is simply to provide the restrictor of the τ-term that ensues by processing the lexical item *if* and the protasis. In that respect there is no independent assertion of the proposition *John cries* which is as it should be according to the desired truth conditions. The formula $Laugh'(Mary')(\tau, s, Cry'(John')(s))$ will be transformed by the Q-Evaluation rule to:

(8.69) $Cry'(John')(\alpha) \rightarrow Laugh'(Mary')(\alpha)$
where $\alpha = \tau, s, [Cry'(John')(s) \rightarrow Laugh'(Mary')(s)]$

According to the semantics proposed in (8.54) above, the *lf* in (8.69) is true in a world w iff all the values of s that satisfy the restrictor, $Cry'(John')$, in w also satisfy the nuclear scope, $Laugh'(Mary')$, in w.

In this configuration a semantic representation containing the connective $\rightarrow$ is derived compositionally at the root node of the main tree without any explicit encoding of it by any NL lexical item.[14] This connective is the same as the one derived by the processing of universally quantifying expressions like *every man* and appears in the logical form by similar means. That is, we assume that material implication is only *part of* the encoded meaning of *if*, the residue being the quantification over situations, so the analysis is richer but not incompatible with the standard material implication interpretations.[15]

This result is achieved in DS in the same way that conjoined propositions are derived from linguistic input containing relative clauses: it is the combination of the lexical and computational actions associated with the parsing of a conditional that derive such universally quantified implicational structures. Indeed, under this analysis, conditional sentences are assumed to derive representations of a similar form as nominal relative clauses, in that a term is shared between the two trees. However, although the main tree in nominal relative constructions provides itself the term to be copied and the LINKed tree provides additional predicative content on that term, here the "relative", the *if*-clause, leads to the construction of a term newly introduced to the main structure. In this respect, the processing of *if*-clauses resembles, in the nominal domain, *headless relatives* in English and other languages like the following:

(8.70) John eats [what(ever) he likes]

(8.71) John will arrest [whoever called]

(8.72)	o Giannis	xeretise	[opion irthe]	*Modern Greek*
	the John	greeted	[who came]	
	John greeted [whoever came]			

In both cases, a term is derived from a secondary tree that is being developed in parallel to the main one. This term is copied to the main tree in order to serve the role of an argument of the predicate in that proposition. As we will see later, the presence of this secondary tree, its attachment to the main structure by the LINK relation and the incorporation of its informational content to the main tree is what explains the subordination features exhibited by *if*-clauses. It is also the reason

[14] However, this is not an analysis where *if* is assigned no meaning and the *if*-clause just provides a restrictor of some other operator cf Kratzer (1986). Hence scopal dependencies with other operators are expected to occur, see e.g. Geurts 2004.

[15] Alternatively, the connective can be interpreted as in Belnap (1970) as argued in Huitink (2008).

that explains the free appearance of *if*-clauses at different positions in the clause as well as their differential contributions to interpretation. We now turn to see the dynamic process that puts all the information together so as to derive the above structure.

8.4.3 The lexical entry for *if*

We will now illustrate the precise route followed by the parser in order to accomplish the construction of a structure such as in (8.68) above from an input string in the form of a conditional sentence. We will start with the instructions stored into the lexical entry for *if*. These are responsible for initiating the building of a new LINKed tree which is going to accommodate the content of the antecedent. Because the result of this process must be a τ-term of type e_s the rules in the lexical entry for *if* initiate structure that is going to derive the required term through the usual structure-building means of DS, i.e., type deduction in parallel with function application. The following is the lexical entry triggered by the parsing of the particle *if*:

(8.73) ***if***

$$
\begin{array}{lll}
\text{IF} & ?Ty(e_s) & (1) \\
\text{THEN} & \texttt{put}(?[\exists \mathbf{x}.Fo(\mathbf{x}) \wedge \langle L \rangle Fo(\mathbf{x})]); & (2) \\
 & \texttt{make}(\langle L \rangle);\ \texttt{go}(\langle L \rangle);\ \texttt{put}(\ ?Ty(e_s)); & (3) \\
 & \texttt{make}(\langle \downarrow_1 \rangle);\ \texttt{go}(\langle \downarrow_1 \rangle); & (4) \\
 & \texttt{put}(Ty(cn_s \rightarrow e_s)), Fo(\lambda P.\tau, P);\ \texttt{go}(\langle \uparrow_1 \rangle); & (5) \\
 & \texttt{make}(\langle \downarrow_0 \rangle);\ \texttt{go}(\langle \downarrow_0 \rangle);\ \texttt{put}(?Ty(cn_s)); & (6) \\
 & \texttt{make}((\langle \downarrow_1 \rangle)); & (7) \\
 & \texttt{put}(Ty(t \rightarrow cn_s)),\ \texttt{freshput}(\mathbf{s}, Fo(\lambda R.\mathbf{s}, R));\ \texttt{go}(\langle \uparrow_1 \rangle); & (8) \\
 & \texttt{make}(\langle \downarrow_0 \rangle);\ \texttt{go}(\langle \downarrow_0 \rangle);\ \texttt{put}(?Ty(t)); & (9) \\
 & \texttt{make}(\langle \downarrow_0 \rangle);\ \texttt{go}(\langle \downarrow_0 \rangle);\ \texttt{put}(Ty(e_s); Fo(\mathbf{s}));\ \texttt{go}(\langle \uparrow_0 \rangle) & (10) \\
\text{ELSE} & \texttt{abort} & (11)
\end{array}
$$

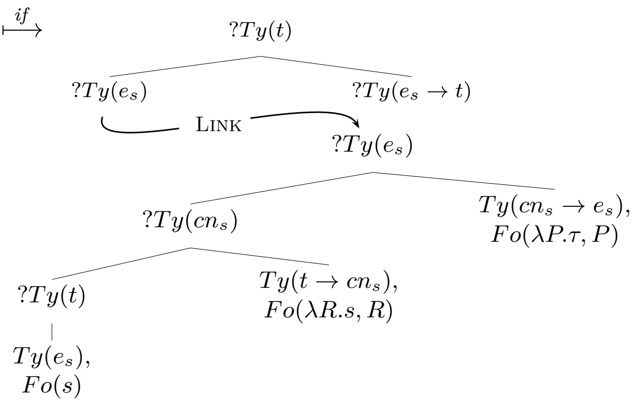
We will illustrate the effect this lexical entry induces by sketching the parsing steps required in processing the string *If John cries Mary laughs*. We start assuming, as above in (8.56), that Introduction and Prediction have created the two nodes lying under the $?Ty(t)$ root node and the pointer now appears at the left argument daughter where there is a requirement for a node of type e_s to be derived ($?Ty(e_s)$):

(8.74) $?Ty(t), \diamondsuit \overset{\text{Introduction}}{\longmapsto} {}^{\text{Prediction}}$

$$
\begin{array}{ccc}
 & ?Ty(t) & \\
?Ty(e_s), \diamondsuit & & ,?Ty(e_s \rightarrow t)
\end{array}
$$

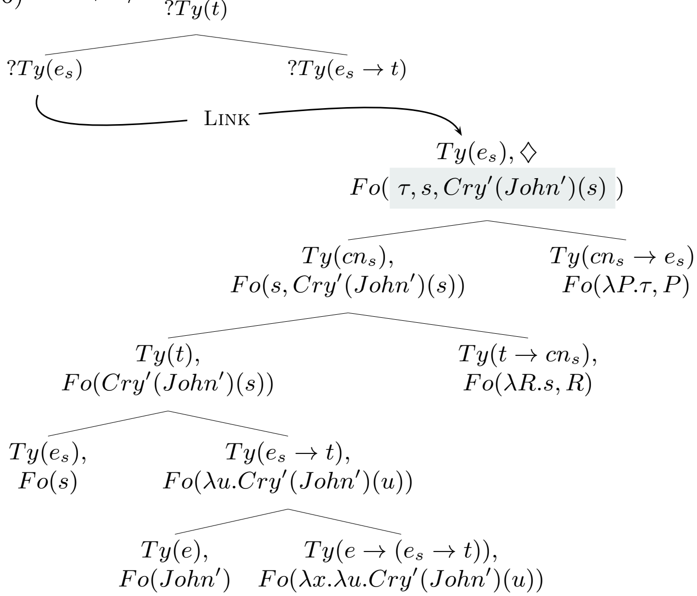
With this being the case, the IF-condition at the lexical entry of *if* is satisfied and the actions specified in the following lines can take place. By line (2) the requirement for a shared formula with a LINKed node is inserted: $?\exists \mathbf{x}.(Fo(\mathbf{x}) \wedge \langle L \rangle Fo(\mathbf{x}))$. The first conjunct of this requirement can only be satisfied by a proper Formula value from the domain D_{Fo}, appearing on the node, i.e., not a metavariable but a proper term. The second conjunct will be satisfied only when the same Formula value appears at a node LINKed to the present one. In combination with the rest of the processing rules, this second part of the requirement will narrow down the choice for a Formula value to just a quantificational term newly derived in the LINKed structure. By the instructions in line (3), $\mathtt{make}(\langle L\rangle);\mathtt{go}(\langle L\rangle);\mathtt{put}(?Ty(e_s))$, a new node connected with the LINK relation to the current one is built and a requirement to derive a term of type e_s is introduced there. This is the node where the derived τ-term of type e_s will eventually appear. Lines (4)-(5) instruct the parser to construct the functor daughter of this node. This is the node where the functor that introduces the τ-binder appears. The sister of this node will provide an argument of $Ty(cn_s)$, the restrictor for the $\tau - term$. This is built by line (6) and a requirement for the appropriate type, a set of situations, is inserted. The structure that is being created here is completely equivalent to a nominal quantificational term restrictor even though the type is sorted to range over situations. At lines (7)-(8) the parser is instructed to introduce another functor daughter which takes as its argument a proposition of type t with a free variable situation argument. The formula value on this node is introduced as follows: $\mathtt{freshput}\ \mathtt{s}, Fo(\lambda R.\mathtt{s}, R))$. The action freshput selects and inserts as (part of) a Formula value for a node the first available variable that has not appeared earlier in the tree being constructed. This ensures that a fresh variable is always introduced. The variable introduced here by the action freshput will be identical to the one introduced as the situation argument of the propositional tree built out of processing the antecedent (this is ensured by the employment of rule-level variables annotated as bold and lower case). The Formula value on this node will bind that variable in order to create a Restrictor of the appropriate shape for the τ-binder at the next level up. By line (9) the propositional tree for the content derived from the antecedent is initiated; this node will eventually be decorated by a formula of type t and will provide the set of situations quantified over. In all respects the tree derived by the antecedent will be a simple predicate-argument structure of type t with the only distinctness being that its situation argument will be necessarily a free variable. This will allow the structure derived to be interpreted as the (description of) a set of situations and thus to serve as the restrictor for a quantificational term:

(8.75) $\overset{if}{\longmapsto}$



Now the regular parsing processes can take over and continue the processing of the antecedent (the fact that the situation accommodating daughter has already been introduced does not affect the operation of Introduction and Prediction). At the parsing stage shown below the antecedent, *if John cries*, has been successfully completed although in terms of interpretation the representation at the type t node is that of a set of situations, not truth-evaluable as a proposition without further ado. The usual rules of function application and type deduction can apply to complete the nodes up to the level at which the link relation terminates:

(8.76) $\overset{if\ John\ cries}{\longmapsto}$



The newly introduced τ-term has as its restrictor the propositional content derived by parsing the string *John cries* under the assumption that a free variable was selected as its situation argument. We now need to copy this term to the main tree. At this point the pointer appears at the top node of the τ-term's subtree. In order to be able to move the pointer back to the main tree and copy the term to its situation node a rule of EVALUATION is required. The rule belongs to the family of rules characterised as LINK EVALUATION rules. It very simply copies the formula from one node to another one which is related to it by means of the LINK relation and simultaneously moves the pointer there. Application of this rule to the tree in (8.76) above will result in the appropriate copying and the requirement at the situation node can now be satisfied:[16]

(8.77) *if John cries* $\overset{\text{LINK EVALUATION}}{\longmapsto}$

$$?Ty(t)$$

$$?Ty(e_s), \cancel{?[\exists \mathbf{x}.Fo(\mathbf{x}) \wedge \langle L\rangle Fo(\mathbf{x})]} \quad Fo(\tau, s, Cry'(John')(s)) \qquad\qquad ?Ty(e_s \rightarrow t)$$

$$\xrightarrow{\text{LINK}}$$

$$Ty(e_s), \; Fo(\tau, s, Cry'(John')(s))$$

$$Ty(cn_s), \; Fo(s, Cry'(John')(s)) \qquad\qquad Ty(cn_s \rightarrow e_s), \; Fo(\lambda P.\tau, P)$$

$$Ty(t), \; Fo(Cry'(John')(s)) \qquad\qquad Ty(t \rightarrow cn_s), \; Fo(\lambda R.s, R)$$

$$Ty(e_s), \; Fo(s) \qquad\qquad Ty(e_s \rightarrow t), \; Fo(\lambda u.Cry'(John')(u))$$

$$Ty(e), \; Fo(John') \qquad\qquad Ty(e \rightarrow (e_s \rightarrow t)), \; Fo(\lambda x.\lambda u.Cry'(John')(u))$$

The parsing of the consequent can now go on as usual deriving the tree shown already in (8.68). The final formula derived at the top node of the global tree will be a universal quantification ranging over (contextually restricted) situations at which John cried and predicating of those that they satisfy the description of Mary's laughing. The QUANTIFIER EVALUATION rules will then derive the appropriate representation in terms of scope and the situation $\tau - term$:

[16]The LINK EVALUATION rule will also introduce a requirement for scope resolution concerning the τ-term (?Sc($\mathbf{x}$)) and an underspecified scope statement at the root node (Scope($\mathbf{U}$<$\mathbf{x}$)).

(8.78) $Ty(t), Fo(Laugh'(Mary', (\boxed{\tau, s, Cry'(John')(s))}) \overset{Q-Eval}{\longmapsto}$
$Fo(Cry'(John')(\alpha) \rightarrow Laugh'(Mary')(\alpha)$
where $\alpha = \tau, s, Cry'(John')(s) \rightarrow Laugh'(Mary')(s))$

Let's see now how the syntactic properties of *if*-clauses are accounted for according to this analysis.

The absence of low construals The parallel properties of *if*-clauses and nominal relatives have led several researchers to the assumption, which is also adopted here, that *if*-clauses are a type of relative clause in the same way that *when*-clauses and *where*-clauses can be taken as relatives over implicit time or location specifications (see e.g. Geis 1985, Bhatt & Pancheva 2001/2005). Nevertheless, there is one difference between these clauses and *if*-clauses. Consider the following:

(8.79) John left when Mary claimed that Bill left

The above is ambiguous between the following two interpretations:

(8.80) John left at the time at which, according to Mary, Bill left

(8.81) John left at the time at which Mary made the following claim: "Bill left"

Unlike *when*-clauses, *if*-clauses are not ambiguous in that respect:

(8.82) John left if Mary claimed that Bill left.

(8.83) In any case that is such that Mary made the claim: "Bill left", John left (in that case)

(8.84) # John left in any one of the circumstances that Bill left according to Mary's claims

When *if*-clauses are analysed as the same type of construction as nominal relatives and *when*-clauses this pattern of idiosyncratic behaviour regarding the strict locality requirement internal to the antecedent is awkward for frameworks that employ uniform processes of movement, or movement-like operations as ad hoc restrictions of the general mechanisms is required in explaining this difference in extractability. In contrast, DS (as well as other lexicalist frameworks) can deal comfortably with idiosyncrasies of this type since, by definition, each word is individually associated with a particular set of procedural actions which, it is plausible, may deviate from their common historical patterns under functional or pragmatic pressures. Hence, in the present analysis, the locality is captured naturally because the initiation of the LINKed tree has been assigned to the actions associated with the word *if* where the situation argument is created immediately as a fixed node on the new tree. In contrast, for *when*-clauses/*where*-clauses, as in nominal relatives, we can assume that the situation argument is constructed by general rules as initially unfixed, hence the possibility of low construal readings.

8.4.4 Conditionals: processing strategies

As we said, terms denoting individuals ($Ty(e_i)$) in DS can be processed by utilising two additional strategies besides their processing in the usual argument positions: either as unfixed nodes initially or finally in the parse or on their own individual subtrees which must be LINKed to another tree of type t. Given that we assume here that *if*-clauses behave as relatives introducing predications on $Ty(e_s)$ arguments we would expect the same strategies to be available to them. First we will examine parsing of *if*-clauses when associated with unfixed nodes and then their contribution to LINKed structures.

8.4.4.1 *If*-clauses as unfixed nodes

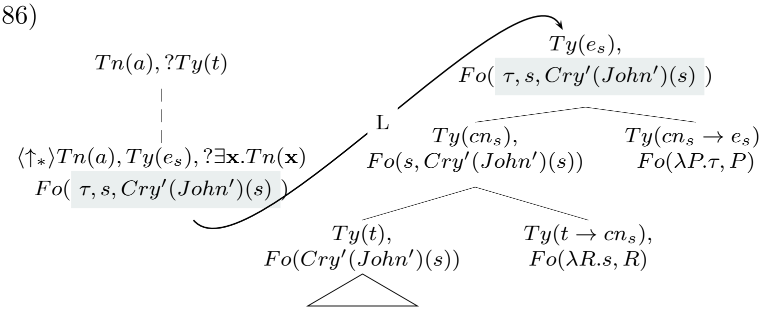
A. Sentence-initial *if*-clause For sentence-initial *if*-clauses we can assume that parsing starts as usual with the $?Ty(t)$ node introduced through the AXIOM and bearing the pointer. As standardly, the rule of *ADJUNCTION, can apply and introduce an unfixed node of type e_s.[17] Diagrammatically we can see the creation of the unfixed node in the following tree which is the initial point of parsing without yet any lexical input:

(8.85) $?Ty(t), \Diamond \quad \overset{*\text{ADJUNCTION}}{\longmapsto} \quad \begin{array}{c} ?Ty(t) \\ \vdots \\ \langle\uparrow_*\rangle, ?Ty(e_s), ?\exists \mathbf{x}.Tn(\mathbf{x}) \end{array}$

As soon as this happens, we are ready to process lexical input since the pointer is in an environment matching what is required for application of the instructions in the *if* lexical entry (see line (1) in (8.73)). According to the instructions given there, a requirement will be then inserted at the unfixed node that a formula value for it must be found at the root of a LINKed tree. The construction of the new LINKed tree will also be initiated and the fresh variable ranging over situations that has been introduced will eventually end up bound by the τ-binder. The rest of the antecedent can then be processed and the LINKed tree will be completed. The *Link Evaluation* rule will copy the term constructed on the unfixed node:

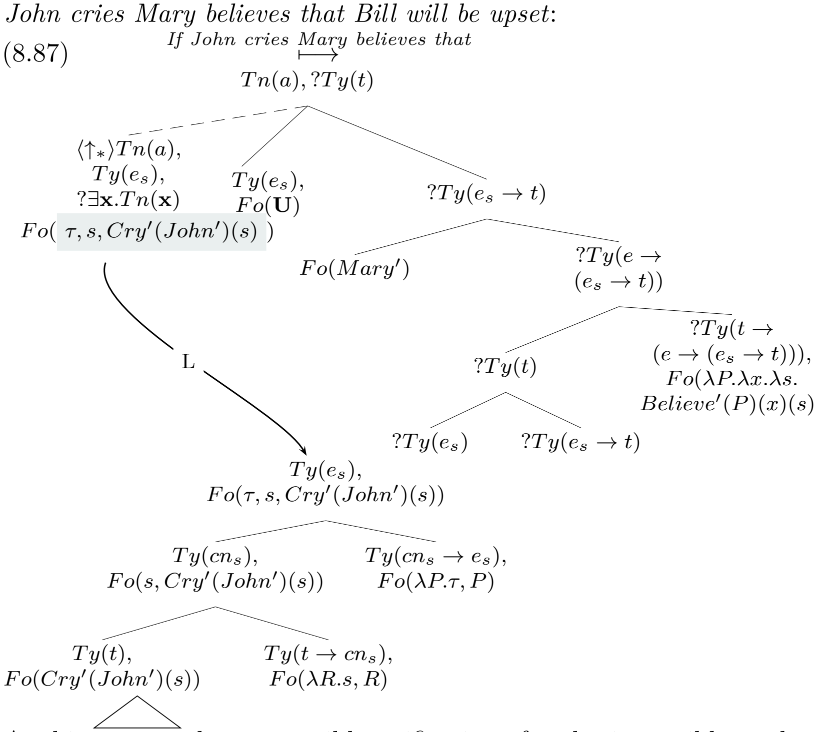
[17] Remember that we have defined type e as being the more general category covering both individual entities and situations.

(8.86)



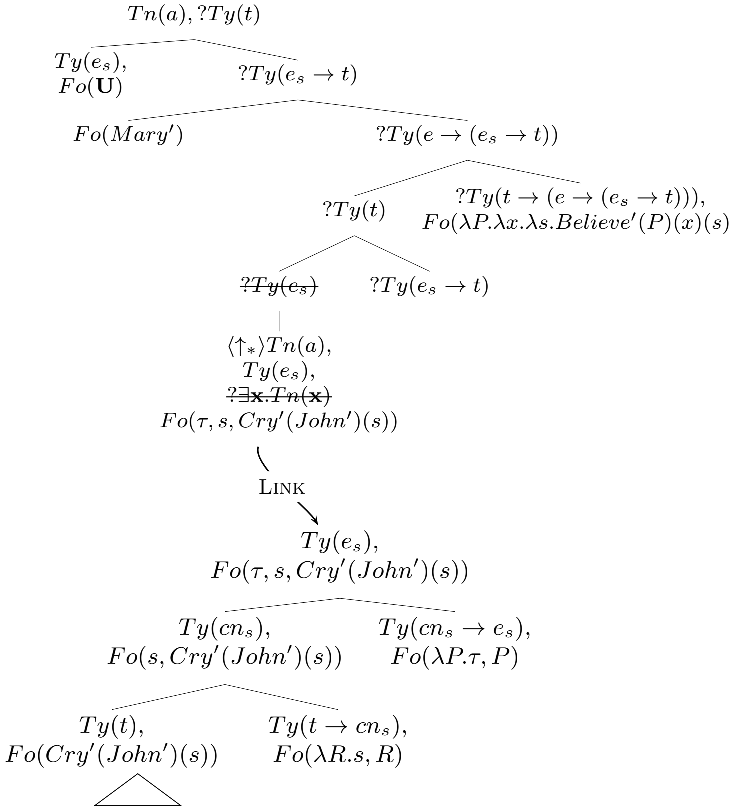
At this point the rule of Completion will move the pointer to the $?Ty(t)$ node and Introduction and Prediction will construct a situation node and its sister. With the pointer at the situation node, Merge can apply to unify it with the unfixed node. Alternatively, Merge, which is an optional operation, will not apply and the parsing will go on constructing more structure with the unfixed node pending. The tree shown below displays a stage of the parsing of a string like *If John cries Mary believes that Bill will be upset*:

(8.87)



At this stage, only one possible unification of nodes is possible so that the pending requirement ($?\exists \mathbf{x}.Tn(\mathbf{x})$) will be satisfied: the unfixed node must merge with the embedded situation node, which will provide a fixed position for it and an appropriate argument for the propositional object of *believe*:

(8.88) $\overset{Merge}{\longmapsto}$



The procedures defined here explain why the processing of such strings might result in two distinct readings, represented by distinct trees according to where Merge has applied:

(8.89) If John cries Sue believes that he just pretends.

(8.89) a. Sue has the following belief: all situations where John cries are situations where he just pretends, or

(8.89) b. all situations where John cries are situations where Sue has the belief that he just pretends.

The two readings are derived in our analysis by assigning to the processing of the sentence two distinct structures. The first reading results when the unfixed situation node that has been created initially and which carries the LINK structure along with it merges with the situation node of the tree created by the processing of the clause which is the object of the verb *believe* . The second reading is derived if the unfixed situation node is merged with the situation node of the tree derived by the processing of the matrix clause.

B. Post-verbal *if*-clause If the antecedent appears post verbally as in (8.90) below

(8.90) John cries if he is upset

then the main tree will have been completed up to the node $Ty(e_s \rightarrow t)$ before processing of the *if*-clause starts:

(8.91) $\overset{John\ \ cries}{\longmapsto}$ $?Ty(t)$

$Ty(e_s)$, $?\exists\mathbf{x}.Fo(\mathbf{x})$ $Fo(\mathbf{U}), \Diamond$

$Ty(e_s \rightarrow t)$, $Fo(\lambda t.Cry'(John')(t))$

$Ty(e)$, $Fo(John')$ $\quad$ $Ty(e \rightarrow (e_s \rightarrow t))$, $Fo(\lambda x.\lambda u.Cry'(x)(u))$

At this stage, the rule of Late-*Adjunction can apply to create a locally unfixed copy of the $Ty(e_s)$ node:

(8.92) $\overset{\text{Late*Adjunction}}{\longmapsto}$ $?Ty(t)$

$Ty(e_s)$, $?\exists\mathbf{x}.Fo(\mathbf{x})$ $Fo(\mathbf{U})$

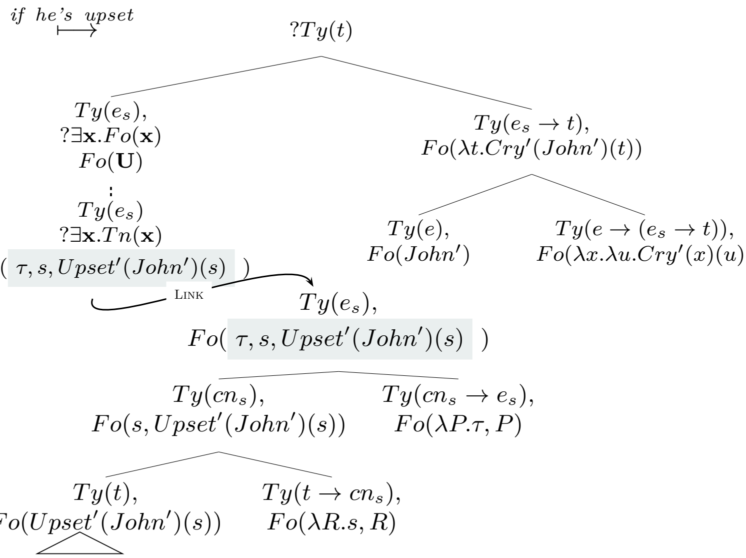
$?Ty(e_s), \Diamond$ $?\exists\mathbf{x}.Tn(\mathbf{x})$

$Ty(e_s \rightarrow t)$, $Fo(\lambda t.Cry'(John')(t))$

$Ty(e)$, $Fo(John')$ $\quad$ $Ty(e \rightarrow (e_s \rightarrow t))$, $Fo(\lambda x.\lambda u.Cry'(x)(u))$

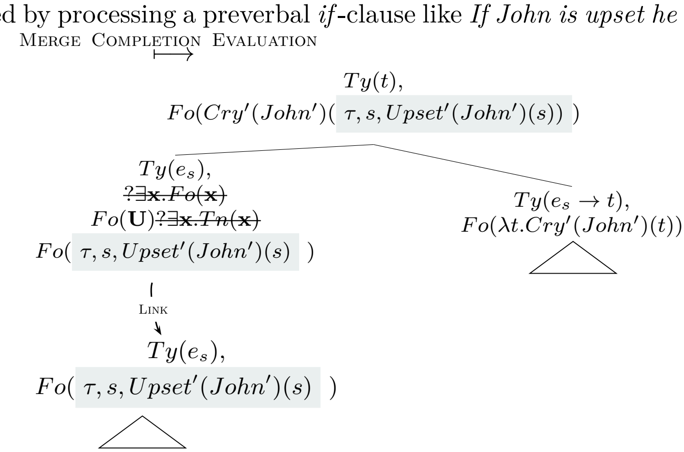
This rule has now created an appropriate context matching what is required for the processing of the word *if* in (8.73). As above, a LINKed tree is constructed from this node and the process and outcome are exactly the same as in the previous cases. The only difference will be that the unfixed node is constructed locally, so that it becomes completed and merges with its mother before any processing of the main tree resumes. Below is the state of the tree after processing of the *if*-clause but just before Merge has occurred:

(8.93)



MERGE can now occur so that both the requirements at the unfixed and fixed nodes can be satisfied. After MERGE, the usual rules will finish the main tree and the result will be indistinguishable to the tree derived by processing a preverbal *if*-clause like *If John is upset he cries*:

(8.94)



Thus, even with post-verbal *if*-clauses, the processing and completion of the tree derived from the antecedent will have been finished before completion of the tree derived from the consequent. This is as it should be since we assume that the antecedent provides a restriction for the situation argument of the main proposition which therefore cannot be completed until that argument is provided. The lexical entry for *if* derives this outcome by allowing the *if*-clause to be processed, if not sentence-initially, only when the pointer is located at the situation argument node of the main tree just before its completion.

The fact that even sentence-initial *if*-clauses need to be able to be processed at unfixed nodes is not required just for the processing of structures like the one in (8.89). As a bonus, this parsing strategy also allows us to explain the apparent cataphoric effects observed in cases of *reconstruction*:

(8.95) [Which of the pictures he$_i$ dislikes] did every professor$_i$ try to hide?

(8.96) If she$_i$'s late again Mary$_i$ will be punished

(8.97) [If pictures of himself$_i$ are on sale], John$_i$ will be happy

As shown in Gregoromichelaki (2006), (8.96)-(8.97) above can be licensed in the same way as (8.95) if the antecedent can be maintained as initially unfixed, which allows delay in the resolution of the pronominals until some appropriate value has been introduced through the processing of the consequent. In addition, this strategy allows quantificational subordination phenomena that occur with conditionals to be accounted for in the same way as the bound variable interpretations we saw in (8.5)-(8.8) earlier:

(8.98) [If her$_i$ child is late from school], every mother$_i$ is upset.

Despite contrary claims in the literature, Gregoromichelaki (2006) has shown that such bindings are possible in either direction, both from the antecedent and from the consequent:[18]

(8.99) If every choice$_i$ is an opportunity, it$_i$'s also a sacrifice

(8.100) ...even if every spaceship$_i$ were made of diamonds, it$_i$ would be cheaper in comparison with its$_i$ virtual price.

(8.101) ..., because if every player$_i$ knew he would be going on the block at the end of every season, he$_i$ would hustle his tail off and make it to every practice,...

The option of parsing *if*-clauses as annotating unfixed nodes allows the modelling of such bindings as naturally following from the parsing strategy selected, even when linearity, quantificational binding and syntactic subordination present conflicting requirements as in (8.98)-(8.101): whether in sentence-initial position or sentence-finally, as we saw, the *if*-clause always provides content that is subordinated to the consequent since it provides a restrictor for one of its arguments that has to be completed first; however, the fact that the node bearing this

[18] For more data and sources see Gregoromichelaki (2006).

content can be processed as unfixed, i.e., as pending until the consequent has been partially completed, allows quantificational elements to be introduced in either of the two LINKed trees with licensed dependencies extending to elements of the other tree (and since scope is dissociated from tree-architecture extra-wide scope is also independent from these requirements).

8.4.4.2 *If*-clauses as LINKed nodes

We have now provided ways of processing preverbal and postverbal antecedents by utilising a single lexical entry for *if* and without introducing any new processing methods to the DS toolbox except the assumption that the situation of evaluation can be processed as a regular (potentially optional) $Ty(e)$ argument. There remains the final challenge of making sense of the heterogeneity apparently indicated by the Haegeman suggestion that there are two wholly different types of *if*-clauses (part of how to resolve this issue has already been discussed above in 8.4.2 with respect to (8.67)). We will now turn to examine whether the rest of the apparatus utilised in DS for the processing of arguments contributed by nominals can be also employed for the processing of conditionals so as to establish an exact parallel between the two types of $Ty(e)$ terms, situations ($Ty(e_s)$) and individuals ($Ty(e_i)$).

A. *If*-clauses in the left periphery *Topicalisation* is the phenomenon of a left-peripheral DP associated with some position more deeply embedded in the following structure:

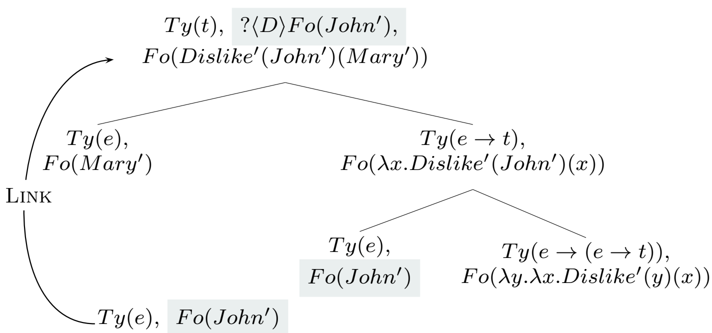
(8.102) John Mary said Bill dislikes ___.

In DS the rule of *ADJUNCTION allows for the initial processing of such left-dislocated DPs as unfixed nodes and the rule of MERGE takes care of their eventual incorporation in the tree. Such constructions contrast with ones that involve anaphoric-like dependencies between left-dislocated phrases and an argument position inside the following structure:

(8.103) (As for) That woman you dislike$_i$, I saw her$_i$ leaving

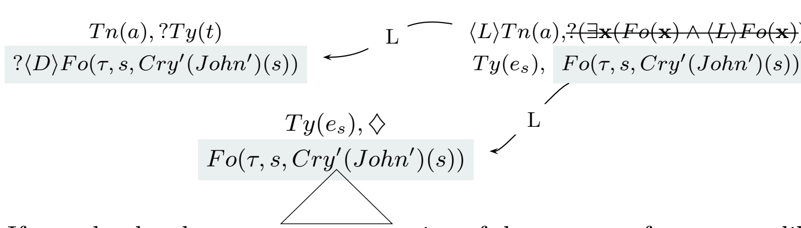
DS licenses such structures by means of the construction of two independent subtrees nodes of which are connected with the LINK relation. Here there is only an obligatory copying relation between the two trees and no merging of nodes. Processing of the left-dislocated phrase is achieved by the TOPIC INTRODUCTION rule (Cann et al 2005). From the AXIOM, the rule allows for the building of a new node required to be of type e. This node is LINKed to the main tree in a sense analogous to it providing the head of a relative clause. Processing of the left-peripheral element can then proceed as usual and after it has been completed a LINK EVALUATION rule is employed to move the pointer to the $?Ty(t)$ node and simultaneously introduce a requirement for a copy of the Formula value appearing at the linked $Ty(e)$ node in some embedded position ($\langle D \rangle$) of the main tree. For illustration consider the tree derived by the processing of such a string:

(8.104) $\xmapsto{(As\ for)\ John\ Mary\ dislikes\ him}$



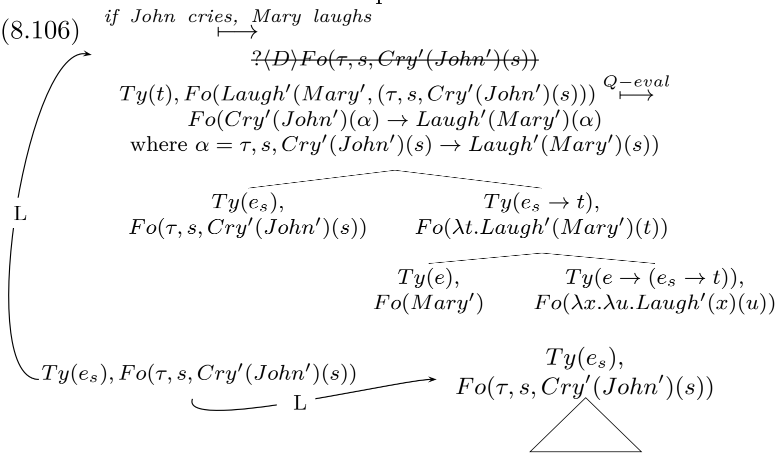
Since the present analysis assumes that processing of *if*-clauses results in the construction of a term of type e it is natural to assume that the same TOPIC INTRODUCTION RULE should be applicable here too. Under this assumption, a situation argument might be constructed independently, externally to the main structure, and a requirement for its Formula value to appear inside the tree representing the consequent can be imposed. The instructions included in the lexical entry for *if* in (8.73) above will be employed unchanged here too to derive a structure similar to that appearing during the processing of nominal topics (see Gregoromichelaki 2006 for full demonstration):

(8.105) $\xmapsto{...\text{TOPIC-INTRODUCTION}\ ...if\ John\ cries...}$



If we take the above as a representation of the content of a sentence like *if John cries, Mary laughs* then the situation metavariable of the main propositional tree must be replaced with the $\tau - term$ derived on the LINKed node otherwise the requirement $?\langle D\rangle Fo(\tau, s, Cry'(John')(s))$ will not be satisfied. By this imposed copying the situation argument of the tree representing the content of the consequent will be unified with the formula value at the top node of the LINKed subtree:

(8.106)



While it seems that there are no evident syntactic differences that distinguish this type of processing conditionals in English[19] it is true that different types of interpretations and inferential effects can be associated with them. The most convincing case for a distinct type of conditional is the *relevance* or *speech act* type, illustrated below:

(8.107) If you are thirsty, there is beer in the fridge.

According to Iatridou (1991), here the antecedent is not used to single out the cases in which the proposition in the consequent is claimed to hold because that proposition is asserted to be true unconditionally. Instead, the antecedent here is taken to specify the circumstances in which it is *relevant*/appropriate to perform the speech act of informing the addressee of the truth of the proposition expressed by the consequent. It has been argued that in such structures there is an implicit performative verb in the main clause which explains the truth conditions. For the example in (8.107) above the underlying structure is claimed to be:

(8.108) If you are thirsty, then [*it is relevant for me to tell you that*] there is beer in the fridge.

One should note that this distinction between *relevance* and other conditionals, in contrast to other distinctions that can be accounted for by purely pragmatic means, has some syntactic basis in V2 languages like German (see e.g. Ebert et al 2008; Koenig and van der Auwera 1988)

[19]Apart from intonational clues which, we assume, are not discreetly encoded in the signal.

and Afrikaans. Native speakers' judgements[20] indicate that it is impossible to sustain the same interpretation between the two conditionals below:

(8.109) *As jy my nodig het,* *ek is by die kantoor* [Afrikaans]
if you me need has, I am at the office
'If you need me, I am at the office'

(8.110) *#As jy my nodig het,* *is ek by die kantoor*
if you me need has, am I at the office
not: If you need me, I am at the office
but: # If you need me, THEN I am at the office

If the antecedent appears as the first element before the verb as in (8.110) then it cannot be assigned a "relevance" interpretation (although it can be assigned a "factual", metarepresentational or echoic interpretation, see (8.67) above). A general account for conditionals needs then to be able to account for why such distinct interpretations are freely available according to context in languages like English while in other languages they seem to have been grammaticised in that they feed or bleed other derivational options. As in V2 languages the *if*-clause can, sometimes, count as the first constituent for V2 (see (8.110)) this necessitates a DS modelling in terms of the unfixed node strategy as this is what underlies the DS characterisation of the V2 phenomenon. We take cases where *if*-clauses appear initially but do not count as the V2 first element as explicit grammaticised indications that they are more "peripheral" to the main structure. This intuition can be given a formal characterisation by taking the *if*-clause as providing content which appears on a node LINKed to the main tree. In such cases, "relevance" interpretations will be preferable (but not obligatory) while with the unfixed node strategy they would be excluded given the information structure requirements associated with this strategy as discussed above in section 8.4 with respect to (8.52)-(8.53).

B. *If*-clauses in the right periphery Dancygier & Sweetser (2005: 174) observe that *if*-clauses (P-clauses in their terminology) appear in the following formal patterns which they assume contrast in terms of interpretations:

(8.111) if P, Q: If the home computer breaks down, I'll work in my office.

(8.112) If P Q: If the home computer breaks down I'll work in my office.

[20] Data from Gregoromichelaki 2006.

(8.113) Q if P: I'll work in my office if the home computer breaks down.

(8.114) Q, if P: I'll work in my office, if the home computer breaks down.

We have now provided an analysis that explains the intuitions discriminating the first three patterns (i.e. LINKed situation node vs. argument situation node). We can now provide a formal basis for the intuition underlying the fourth pattern, i.e. that it indicates that the P-clause relies on prior contextual justification of its content:

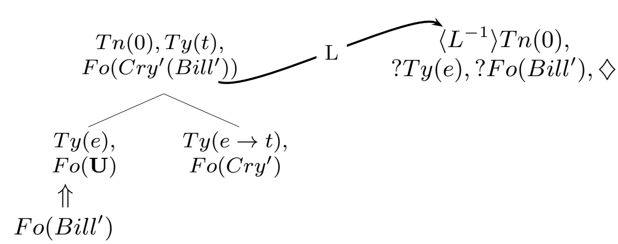
> The Q-clause is then followed (after a comma/pause) by P, which either further restricts the context in which the assertion of Q is valid, or justifies the communication of Q as appropriate (this may involve conditions on speech acts and metalinguistic conditions) (Dancygier & Sweetser ibid: 175).

In DS a similar pattern as regards nominals has been identified (Cann et al 2005, Ch. 5):

(8.115) He cries, Bill

A right peripheral nominal expression can be presented with an anaphoric expression inside the main clause necessarily identified as co-referential with it. The DS parsing strategy that handles this pattern employs the rule of RECAPITULATION (Cann et al. 2005). According to this, following the completion of a propositional type t structure, the rule initiates a LINKed tree which bears the requirement to include a formula value identical to one appearing inside the tree from which the LINK relation originates. Since the latter has necessarily been completed prior to the application of the rule, the formula value that needs to be copied must have already been provided contextually. As an illustration consider the (schematic) tree that will be derived for the string in (8.115) (where the double arrow indicates SUBSTITUTION of a metavariable formula value on a treenode by a contextually provided value):

(8.116)



We can give a parallel analysis for the pattern in (8.114). The right peripheral *if*-clause can provide justification for the selection of the contextually derived situation term or further restrict it by means of providing additional clues for the value of the metavariable $\boldsymbol{R}$, the contextual restriction always included in such terms (see section 8.4.2). As an illustration consider the tree derived from the Q-clause in (8.114) after application of the Recapitulation rule (now taking the type e mentioned in this rule as the general supertype). The contextual provision of a value for the metavariable at the situation argument node and the requirement for it to appear again in the new $?Ty(e_s)$ tree initiated are shown. Parsing of the *if*-clause provides the required value. After processing of the P-clause has been finalised the (schematic) structure that ensues for (8.114) above is as follows:

(8.117)

$$Tn(0), Ty(t),$$
$$Fo(Work'\text{-}in\text{-}office'(Speaker')(\,\tau, s, Break\text{-}down'(Computer)(s)\,))$$

$$Ty(e_s),\quad Fo(\mathbf{S}) \qquad\qquad Ty(e_s \to t),\quad Fo(\lambda v.Work\text{-}in\text{-}office'(Speaker')(v))$$
$$\Uparrow$$
$$Fo(\,\tau, s, Break\text{-}down'(Computer)(s)\,)$$

L

$$\langle L^{-1}\rangle Tn(0),$$
$$Ty(e_s),\ \cancel{?Fo(\tau, s, Break\text{-}down'(Computer)(s))},$$
$$Fo(\,\tau, s, Break\text{-}down'(Computer)(s)\,)$$

L

$$Ty(e_s),$$
$$Fo(\,\tau, s, Break\text{-}down'(Computer)(s)\,)$$

8.5 Conclusion

There are multiple analyses of conditionals which attempt to classify their disparate interpretations (see e.g. Declerck & Reed (2001) who identify a multitude of semantics) and ground those on the basis of underlying structural ambiguities. In contrast, an account in the spirit of DS ought to allow such a multiplicity of interpretations not as directly encoded in the linguistic system but as the effect of the inherent context dependence of the lexical/structural resources and the processing strategies available. In accordance to this, the aim of the present account was to provide the minimal skeletal form of the truth conditions of conditionals which (a) further processes of reasoning and pragmatic inferencing will take as input in order to derive the requisite interpretations (see Björnsson and Gregoromichelaki in prep) and (b) provide the source of providing answers to puzzles in the syntax/semantics interface. Given that DS is more parsimonious in terms of representational levels, eschewing the postulation of a level of hierarchical organisation over strings, while at the same time allowing the compositional derivation of the requisite semantic representation, this account is preferable as it covers all the syntactic facts (see, e.g. sect 8.3.1) via independently motivated processing considerations. In terms of interpretational effects, the review of the data leads to the conclusion that there is a distinction between peripheral and more integrated *if*-clauses, a fact reflected here in their processing, via identical syntactic assumptions, as either annotating a LINKed subtree or an unfixed node. Unlike a structural ambiguity account, this is a natural explanation of the distinctions observed as variation in the time-linear presentation of truth-conditionally equivalent forms reflects differences in information status, a well-known factor that underlies the existence of variable processing methods in general. Moreover, both from a functional and a psycholinguistic point of view, the availability of alternative strategies is an essential component of NL processing as it both facilitates incremental processing (Ferreira 1996) and allows the structuring of expressions according to the gradual contextual availability of information.