

Modelling Dialogue and Language Change: a Dynamic Syntax account

Ruth Kempson Miriam Bouzouita Ronnie Cann
King's College London, University of Edinburgh

{ruth.kempson, miriam.bouzouita}@kcl.ac.uk, r.cann@ed.ac.uk
<http://www.kcl.ac.uk/research/groups/ds/essllicourse.html>

Acknowledgements:

Eleni Gregoromichelaki, Yo Sato, Chris Howes, Jieun Kiaer,
Andrew Gargett, Stelios Hagikyriakides, Pat Healey, Greg Mills, Duilio D'Alfonso..

August 1, 2008

Language as an Evolving Dynamic System

- Why model grammars with built-in parsing dynamics?
- Why model conversational dialogue?
- What is the context in context-dependent processing?
- How can we reflect the evolving nature of syntactic change?
- How can we explain apparently blind syntactic processes?
- How do morphological opacities arise?

Why Model Grammars with Built-in Parsing Dynamics?

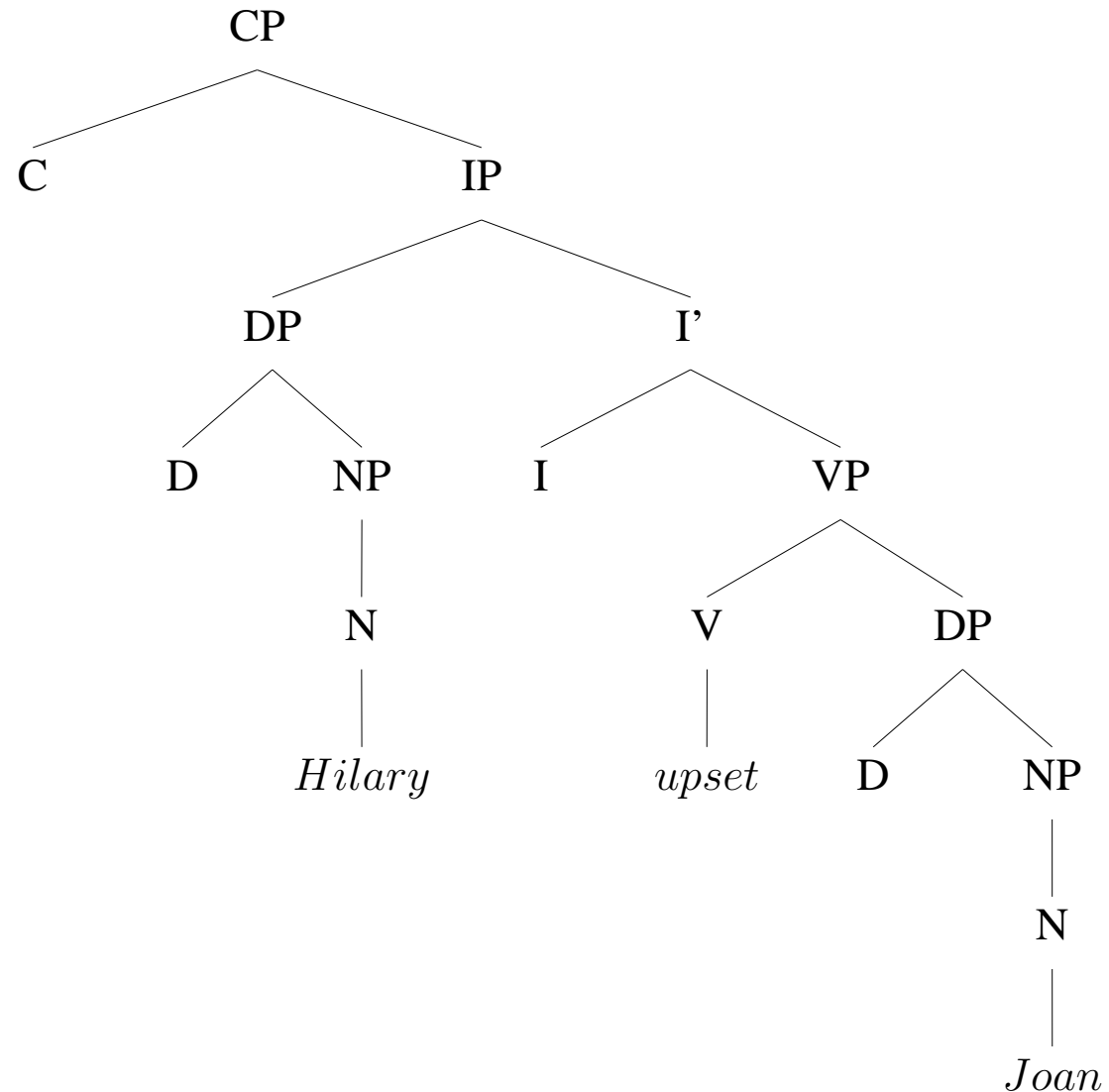
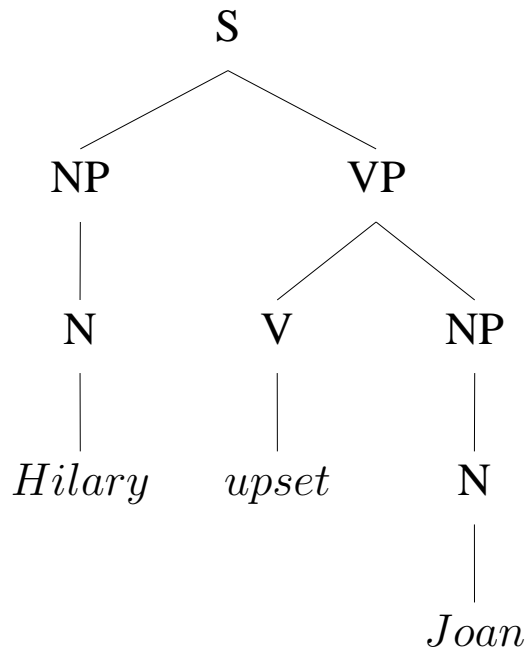
- Making sense of progressive syntactic change
- Syntax as the architecture for parsing in context – Lecture I
- Interaction of Syntax and Anaphora Lecture II
 - (i) relative clause construal
 - (ii) periphery effects
 - (iii) free word order effects and scrambling
- Ellipsis and Dialogue – Lecture III
 - (i) context as record of construction PROCESS
 - (ii) tight coordination of parsing and production
 - (iii) parallelism and priming effects explained
- Syntactic change via routinisation of alternatives (w. Miriam Bouzouita)
 - (i) Re-analyses from overlapping alternatives underpin
Medieval-Modern Spanish clitic placement shift – Lecture IV
 - (ii) Morpho-syntactic ‘gaps in paradigm’ explained:
The Person Case Constraint – Lecture V
- Language competence as a capacity for context-dependent processing

Lecture I: contents

- (1) The concept of structure
- (2) Describing tree growth
- (3) Tree growth actions
- (4) Lexical actions: the procedural language
- (5) Defining as tree growth procedures:
 - long-distance dependency
 - case specifications
 - anaphora construal
- (6) Cross-language variation in argument optionality
- (7) Wellformedness

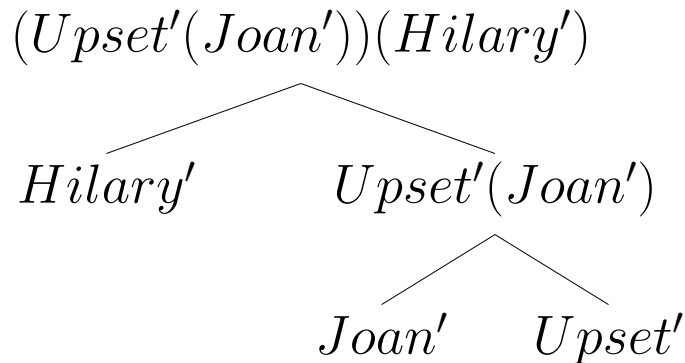
Syntax as structure inhabited by strings?

(1) Most Grammar formalisms characterise properties of strings of words:



Syntax as mechanisms for constructing representations of content?

(2) In Dynamic Syntax, what is ultimately characterised is Propositional Structure, represented as tree structure:

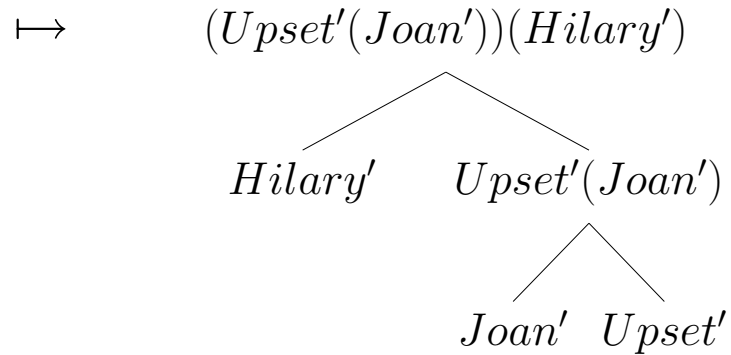
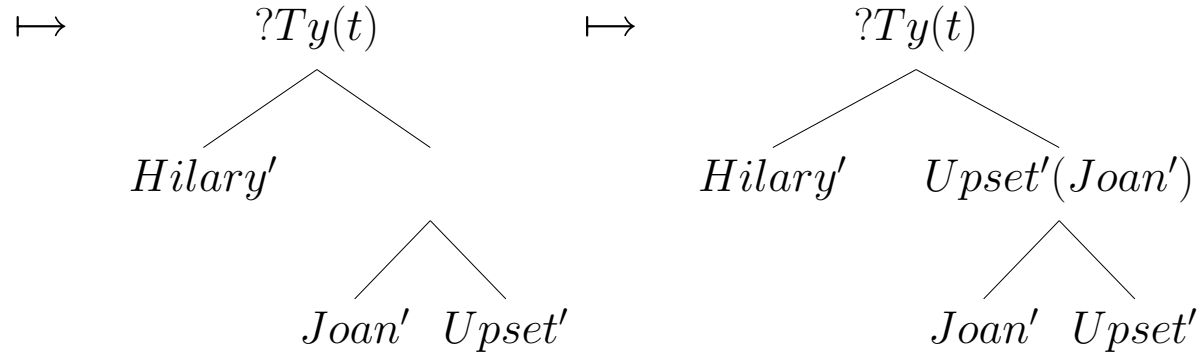
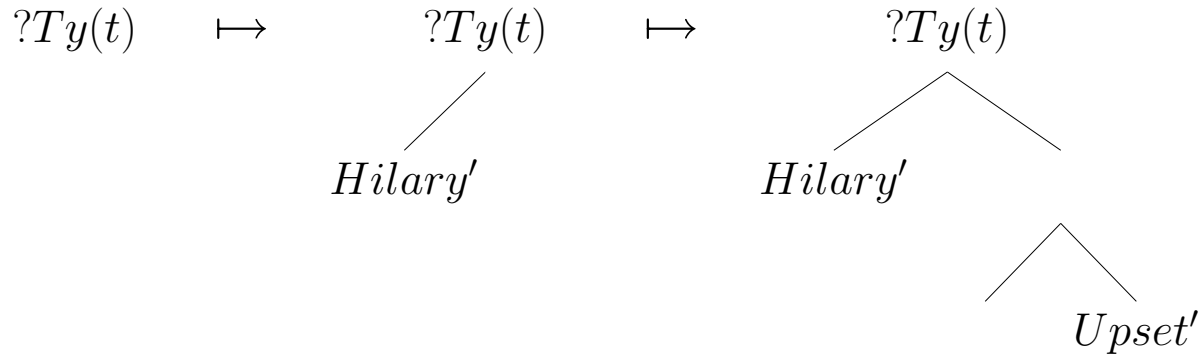


DS is concerned with characterising the GROWTH of such representations through parsing strings uttered in contexts.

(3) The syntactic process:

- the progressive enrichment of some partial structure through the parse of a string of words in context;
- information is built up on a left-to-right, word-by-word basis relative to some context against which choices may be made as the construction process proceeds.

Dynamic syntax: parsing *Hilary upset Joan*



The Tools of Dynamic Syntax

- (4) We have a two-fold challenge ahead.
- to set out a model of how interpretation is recovered in context;
 - to establish why this constitutes the basis for syntactic explanations.

Vocabulary for labelling partial trees

Treenodes are decorated by DECLARATIVE UNITS which are consistent sets of LABELS.

- (5) DECLARATIVE UNIT (DU): $\{La_1(\alpha), \dots, La_n(\omega)\}$
 where $\forall i, j (1 \leq i \leq n) [i = j \wedge La_i(\beta) \wedge La_j(\gamma)] \rightarrow \beta \leq \gamma$
- (6) FORMULAE:

Fo. (Fo(Sing'(John')), Fo(John'), Fo(Sing')).

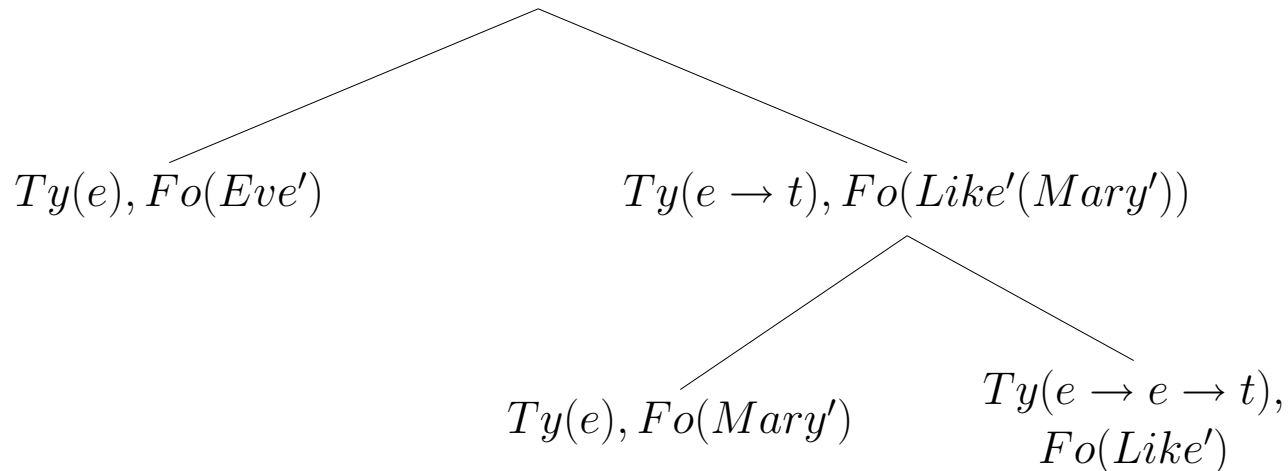
The Tools of Dynamic Syntax

(7) TYPE

| Type | Description | Example expression |
|---|--------------------------------|--|
| $Ty(e)$ | Individual term | $Fo(Mary')$, $Fo(\epsilon, x, Student(x))$ |
| $Ty(t)$ | Proposition | $Fo(Sing'(John'))$, $Fo((Upset'(Hilary'))(Joan'))$ |
| $Ty(e \rightarrow t)$ | (1-place) Predicate | $Fo(Upset'(Hilary'))$, $Fo(Run')$ |
| $Ty(e \rightarrow (e \rightarrow t))$ | (2-place) Predicate | $Fo(Upset')$, $Fo(Give'(John'))$ |
| $Ty(e \rightarrow (e \rightarrow (e \rightarrow t)))$ | (3-place) Predicate | $Fo(Give')$, $Fo(Put')$ |
| $Ty(t \rightarrow (e \rightarrow t))$ | (Proposition taking) Predicate | $Fo(Believe')$, $Fo(Say')$ |
| $Ty(cn)$ | Nominal | $Fo(x, Student'(x))$, $Fo(y, Father(John)(y))$ |
| $Ty(cn \rightarrow e)$ | Quantifier | $Fo(\lambda P.\epsilon, P)$, $Fo(\lambda P.\tau, P)$ |

The Tools of Dynamic Syntax: using semantic concepts as syntax

(8) $Ty(t), Fo((Like'(Mary'))(Eve'))$



(9) REQUIREMENTS: goals to be undertaken with respect to providing an instance of some label at the current node

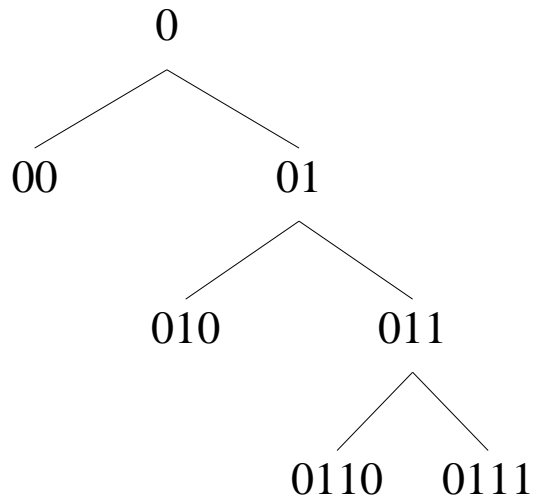
- a. $Ty(t)$ holding at a node means that some formula of type t has been constructed at that node;
- b. $?Ty(t)$ holding at a node shows that all that has been established is a **goal** of constructing such a formula.

(10) \diamond : the “pointer” indicates the task state currently under development.

AXIOM: $Tn(0), ?Ty(t), \diamond$

The Logic of Trees: identifying nodes in a tree

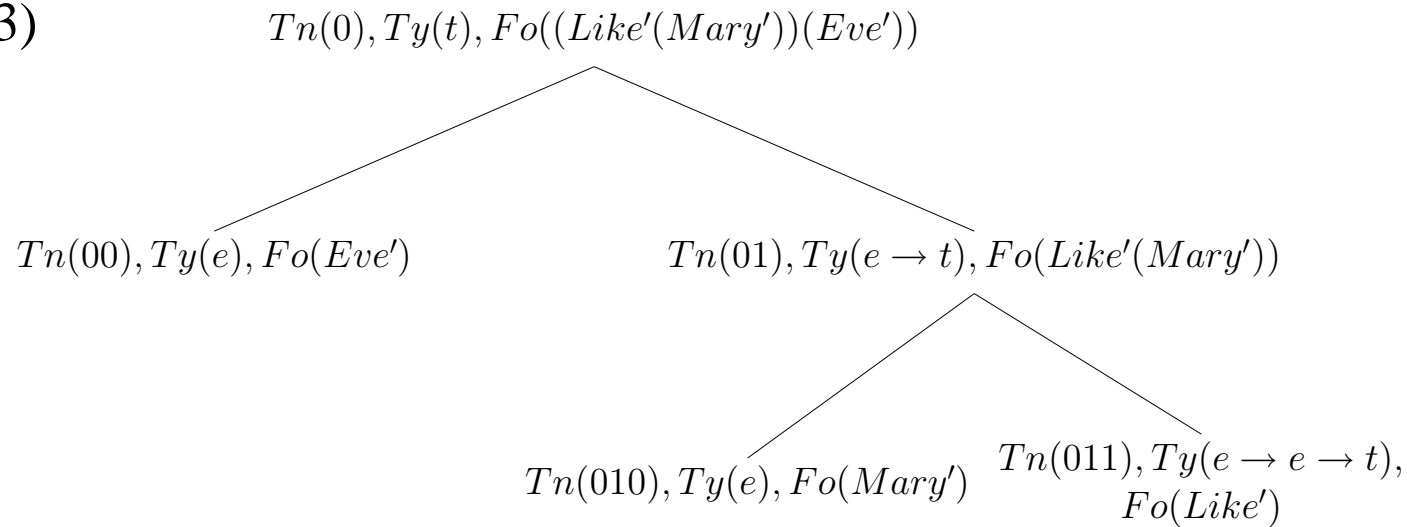
(11)



(12)

- a. $Tn(0)$: topnode.
- b. $Tn(n0)$: argument daughter of $Tn(n)$.
- c. $Tn(n1)$: functor daughter of $Tn(n)$.

(13)



The Logic of Trees: Logic of Finite Trees

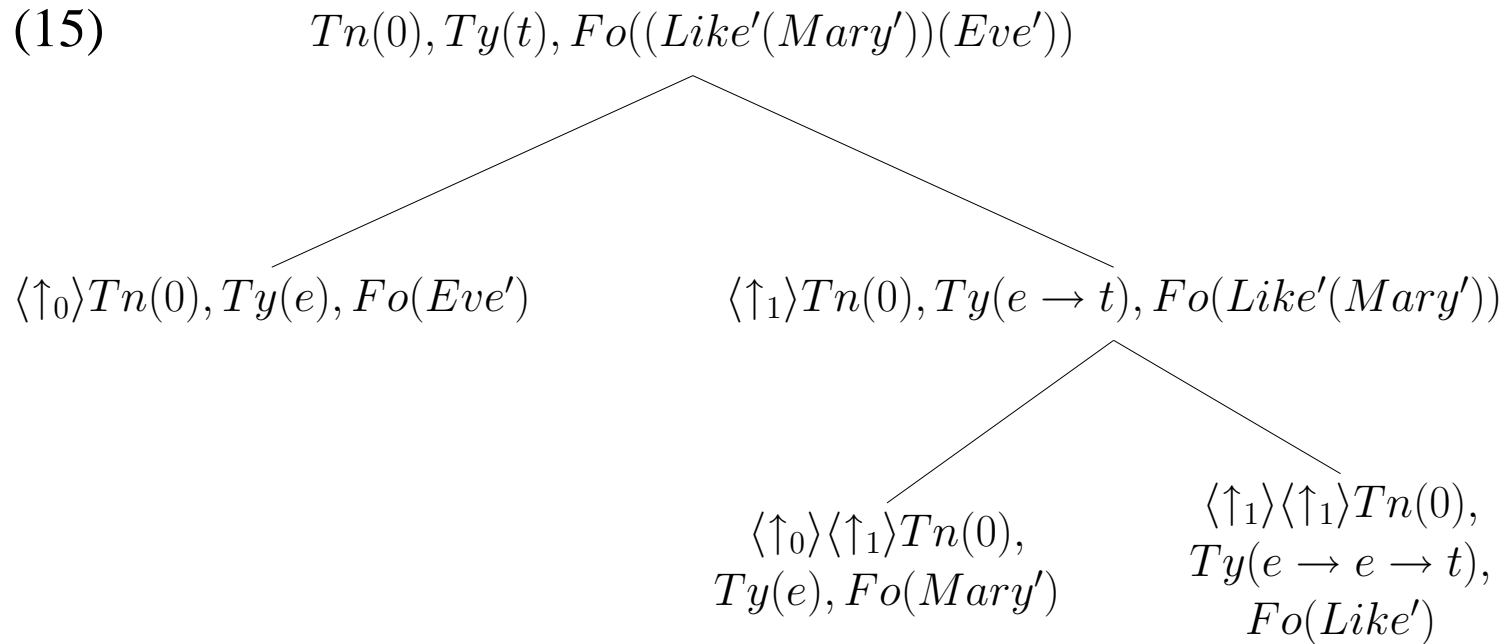
(14) Two basic modalities:

- a. $\langle \downarrow \rangle$ ‘down’: the DAUGHTER RELATION;
- b. $\langle \uparrow \rangle$ ‘up’: the MOTHER RELATION.

From node n :

| | |
|----------------------------------|---|
| $\langle \downarrow_0 \rangle X$ | X holds at the argument daughter of n |
| $\langle \downarrow_1 \rangle X$ | X holds at the functor daughter of n |
| $\langle \uparrow \rangle X$ | X holds at the mother of n |
| $\langle \downarrow_* \rangle X$ | X holds at a node dominated by n |
| $\langle \uparrow_* \rangle X$ | X holds at a node that dominates n |
| $\langle L \rangle X$ | X holds at a node that is linked to n |
| $\langle L^{-1} \rangle X$ | X holds at a node that n is linked to |
| $\langle D \rangle X$ | X holds at a node globally dominated by n |
| $\langle U \rangle X$ | X holds at a node that globally dominates n |

The Logic of Trees: identifying nodes relative to the root



Nodes are uniquely identified by their relations to other nodes in the tree:
each has a uniquely characterised position in the configuration.

The Logic of Trees: preliminaries for tree growth

(16) **Modal requirements:** Modal operators may also be used in conjunction with the notion of requirement to constrain the development of the tree.

a. $\langle \downarrow_0 \rangle Fo(\alpha)$

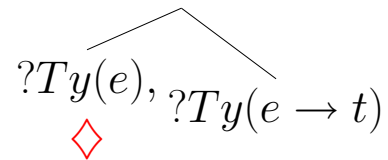
indicates that the formula value α holds of my argument daughter;

b. $? \langle \downarrow_0 \rangle Fo(\alpha)$

states that *at some point in constructing the current tree*, $Fo(\alpha)$ must decorate my argument daughter.

One opening strategy in building up trees is the anticipation of subject and predicate nodes by imposing two modal requirements, one for a one-place predicate node, and one for an argument node to go with it:

$$?Ty(t), ? \langle \downarrow_0 \rangle Ty(e), ? \langle \downarrow_1 \rangle Ty(e \rightarrow t)$$



(17) Case specifications as output filters. Nominative case (English):

$$? \langle \uparrow_0 \rangle (Ty(t) \wedge \exists x. Tns(x)). \text{ Accusative case:}$$

$$? \langle \uparrow_0 \rangle Ty(e \rightarrow t)$$

4 Constructing Trees: 4.1 Computational Actions

(18) TRANSITION RULES

a.
$$\frac{\text{Input Tree Description}}{\text{Output Tree Description}}$$

b.

$$\frac{\{\dots \phi \dots \diamond\}}{\{\dots \psi \dots \diamond \dots\}}$$

(19) INTRODUCTION: adds requirements for daughter nodes of certain types.

Rule:

$$\frac{\{\dots ?Ty(Y) \dots \diamond\}}{\{\dots ?Ty(Y), ?\langle \downarrow_0 \rangle Ty(X), ?\langle \downarrow_1 \rangle Ty(X \rightarrow Y), \dots \diamond\}}$$

Tree growth:

$$?Ty(X), \diamond \quad \mapsto \quad ?Ty(X), ?\langle \downarrow_0 \rangle Ty(Y), ?\langle \downarrow_1 \rangle Ty(Y \rightarrow X), \diamond$$

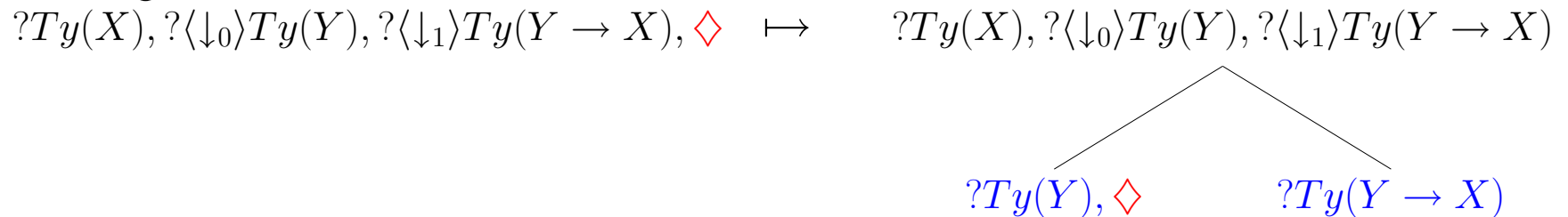
Computational Actions

(20) PREDICTION: builds daughter nodes with type requirements

Rule:

$$\frac{\{Tn(n), \dots, ?\langle \downarrow_0 \rangle \phi, ?\langle \downarrow_1 \rangle \psi, \diamond\}}{\{Tn(n), \dots, ?\langle \downarrow_0 \rangle \phi, ?\langle \downarrow_1 \rangle \psi\}, \{\langle \uparrow_0 \rangle Tn(n), ?\phi, \diamond\} \{\langle \uparrow_1 \rangle Tn(n), ?\psi\}}$$

Tree growth:



Computational Actions

(21) SUBJECT AND PREDICATE

a. Introduction:

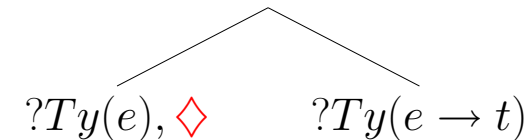
$$\frac{\{Tn(n), ?Ty(t), \diamond\}}{\{Tn(n), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond\}}$$

b. Prediction:

$$\frac{\{\{Tn(n), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond\}\}}{\{\{Tn(n), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t)\}, \{\langle \uparrow_0 \rangle Tn(n), ?Ty(e), \diamond\}, \{\langle \uparrow_1 \rangle Tn(n), ?Ty(e \rightarrow t)\}\}}$$

c. Introduction and Prediction of Subject and Predicate (tree growth)

$$?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond \mapsto ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t)$$



4.2 Lexical Information: Procedures for tree growth

(22) LEXICAL ACTIONS

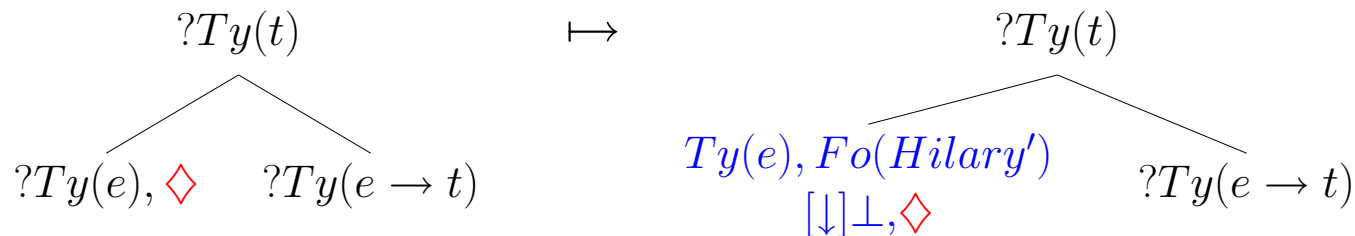
- a. `make(...)` which makes a new node;
- b. `go(...)` which moves the pointer to the node specified in the value;
- c. `put(...)` which annotates a node with certain information.

(23) FORMAT OF LEXICAL ENTRIES

| | | |
|------|--------|---------------------|
| IF | ?Ty(X) | Trigger |
| THEN | ... | Actions |
| ELSE | ... | Elsewhere Statement |

| | | | |
|--------------------|------|-------------------------------|------------|
| (24) <i>Hilary</i> | IF | ?Ty(e) | Trigger |
| | THEN | put(Ty(e), Fo(Hilary'), [↓]⊥) | Annotation |
| | ELSE | ABORT | Failure |

(25) Parsing *Hilary*:



4.2 Lexical Information

(26) $[\downarrow]\perp$: “the bottom restriction”, a terminal node marker: “necessarily below the current node (for every node the current node immediately dominates), the falsum holds”.

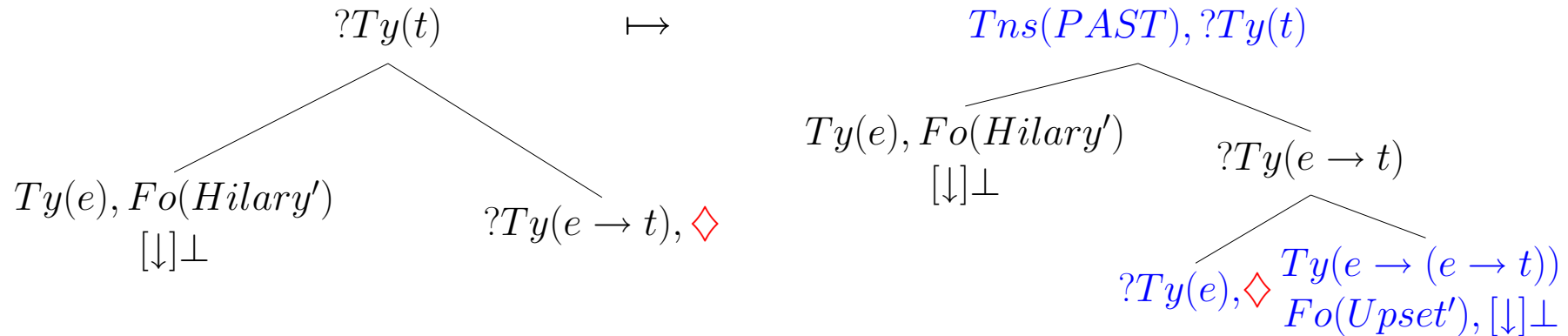
(27) *upset*

| | |
|---|--|
| <p>IF $?Ty(e \rightarrow t)$</p> <p>THEN $go(\langle \uparrow_1 \rangle ?Ty(t)),$</p> <p style="padding-left: 20px;">$put(Tns(PAST)),$</p> <p style="padding-left: 20px;">$go(\langle \downarrow_1 \rangle ?Ty(e \rightarrow t)),$</p> <p style="padding-left: 20px;">$make(\langle \downarrow_1 \rangle), go(\langle \downarrow_1 \rangle)$</p> <p style="padding-left: 20px;">$put(Fo(Upset'), Ty(e \rightarrow (e \rightarrow t), [\downarrow]\perp);$</p> <p style="padding-left: 20px;">$go(\langle \uparrow_1 \rangle),$</p> <p style="padding-left: 20px;">$make(\langle \downarrow_0 \rangle);$</p> <p style="padding-left: 20px;">$go(\langle \downarrow_0 \rangle);$</p> <p style="padding-left: 20px;">$put(?Ty(e))$</p> <p>ELSE ABORT</p> | <p>Predicate trigger</p> <p>Go to propositional node</p> <p>Tense information</p> <p>Go to predicate node</p> <p>Make functor node</p> <p>Annotation</p> <p>Go to mother node</p> <p>Make argument node</p> <p>Go to argument node</p> <p>Annotation</p> |
|---|--|

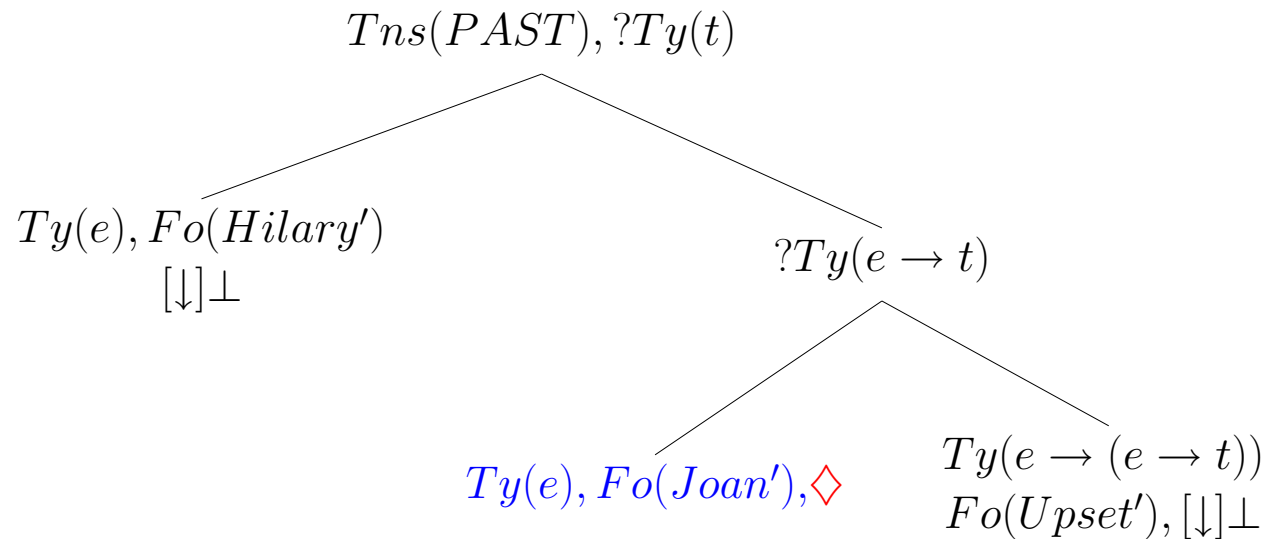
This format is ineliminable. Computational actions are also expressible in this procedural notation.

4.2 Lexical Information

(28) Parsing *Hilary upset*



(29) Parsing *Hilary upset Joan*



4.3 Completing the Tree

(30) THINNING:

eliminates requirements

$$\frac{\{\dots\phi\dots?\phi\dots, \diamond\}}{\{\dots\phi\dots, \diamond\}}$$

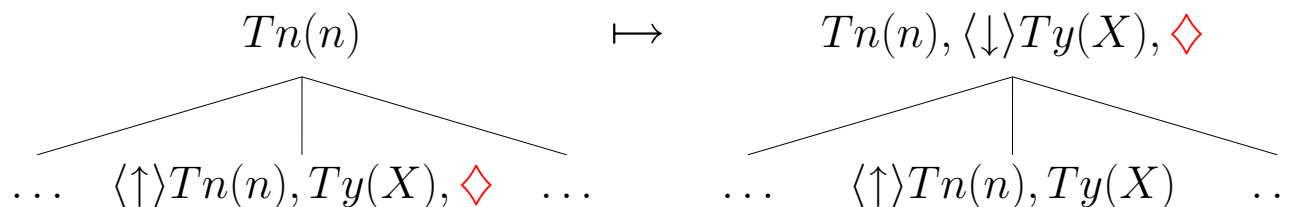
(31) COMPLETION:

moves pointer to mother and copies information about daughter node

Rule:

$$\frac{\{Tn(n)\dots\}, \{\langle\uparrow_i\rangle Tn(n), \dots, Ty(X), \dots, \diamond\}}{\{Tn(n), \dots, \langle\downarrow_i\rangle Ty(X), \dots, \diamond\} \{ \langle\uparrow_i\rangle Tn(n), \dots, Ty(X), \dots, \} } \\ i \in \{0, 1, *\}$$

Tree growth:



4.3 Completing the Tree

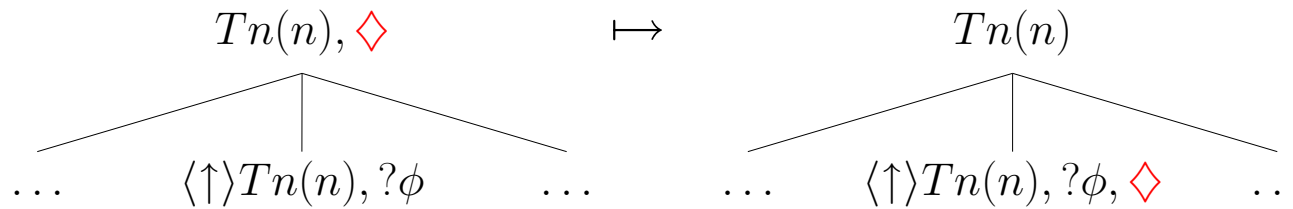
(32) ANTICIPATION:

moves pointer down to a daughter node that contains a requirement

Rule:

$$\frac{\{Tn(n), \dots, \diamond\}, \{\langle \uparrow \rangle Tn(n), \dots, ?\phi, \dots\}}{\{Tn(n), \dots\} \{ \langle \uparrow \rangle Tn(n), \dots, ?\phi, \dots, \diamond \}}$$

Tree growth:



4.3 Completing the Tree

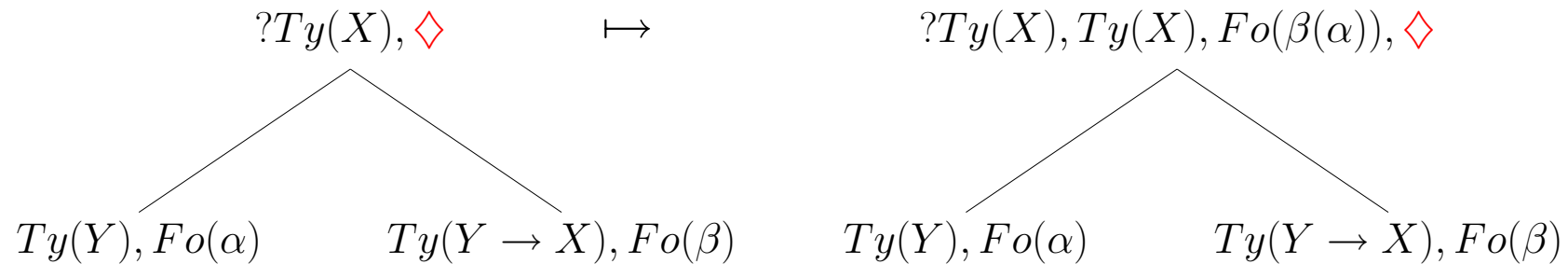
(33) ELIMINATION:

performs functional application

Rule:

$$\frac{\{ \dots \langle \downarrow_0 \rangle (Fo(\alpha), Ty(Y)), \langle \downarrow_1 \rangle (Fo(\beta), Ty(Y \rightarrow X)) \dots, \diamond \}}{\{ \dots Fo(\beta(\alpha)), Ty(X), \langle \downarrow_0 \rangle (Fo(\alpha), Ty(Y)), \langle \downarrow_1 \rangle (Fo(\beta), Ty(Y \rightarrow X)) \dots, \diamond \}}$$

Tree growth:



Parsing *Hilary upset Joan*

$Tn(n), ?Ty(t), \diamond$

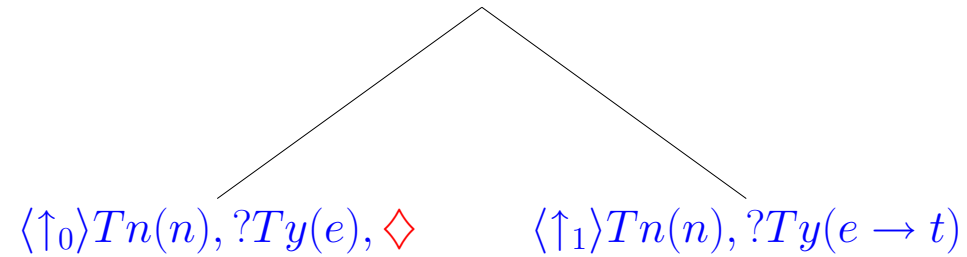
AXIOM

Parsing Hilary upset Joan

$Tn(n), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond$

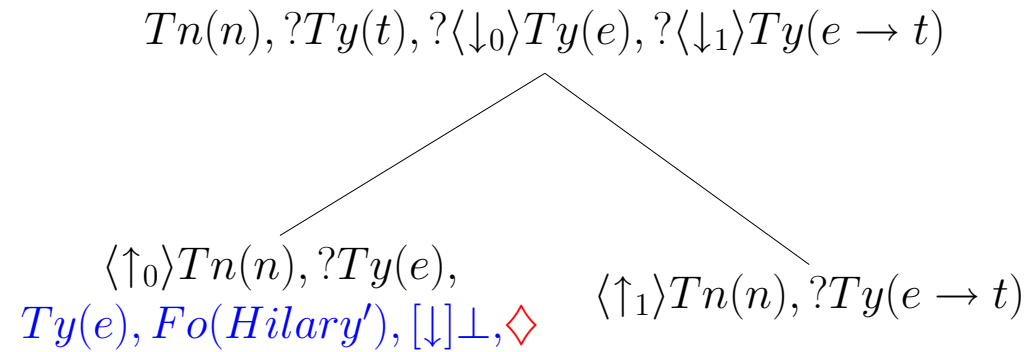
INTRODUCTION (Subject/Predicate)

Parsing *Hilary upset Joan*

$$Tn(n), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t)$$


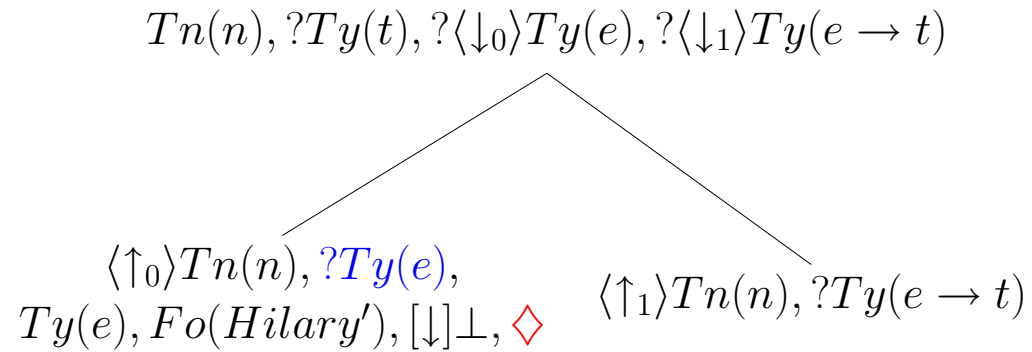
PREDICTION (Subject/Predicate)

Parsing Hilary upset Joan



Hilary

Parsing *Hilary upset Joan*



Thinning

Parsing Hilary upset Joan

$$Tn(n), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), \langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond$$

$$\langle \uparrow_0 \rangle Tn(n), Ty(e), \\ Fo(Hilary'), [\downarrow] \perp$$

$$\langle \uparrow_1 \rangle Tn(n), ?Ty(e \rightarrow t)$$

Completion

Parsing *Hilary upset Joan*

$Tn(n), ?Ty(t), \langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond$

$\langle \uparrow_0 \rangle Tn(n), Ty(e),$
 $Fo(Hilary'), [\downarrow] \perp$

$\langle \uparrow_1 \rangle Tn(n), ?Ty(e \rightarrow t)$

Thinning

Parsing *Hilary upset Joan*

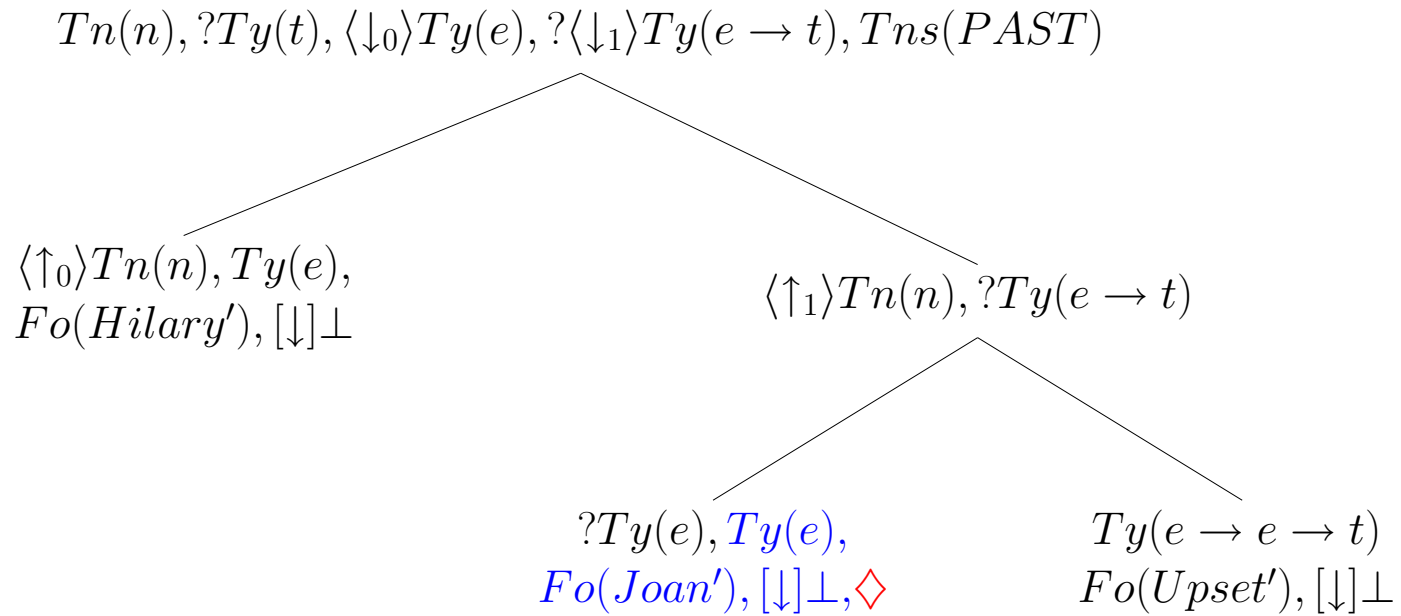
$$Tn(n), ?Ty(t), \langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t)$$

$$\langle \uparrow_0 \rangle Tn(n), Ty(e), \\ Fo(Hilary'), [\downarrow] \perp$$

$$\langle \uparrow_1 \rangle Tn(n), ?Ty(e \rightarrow t), \diamond$$

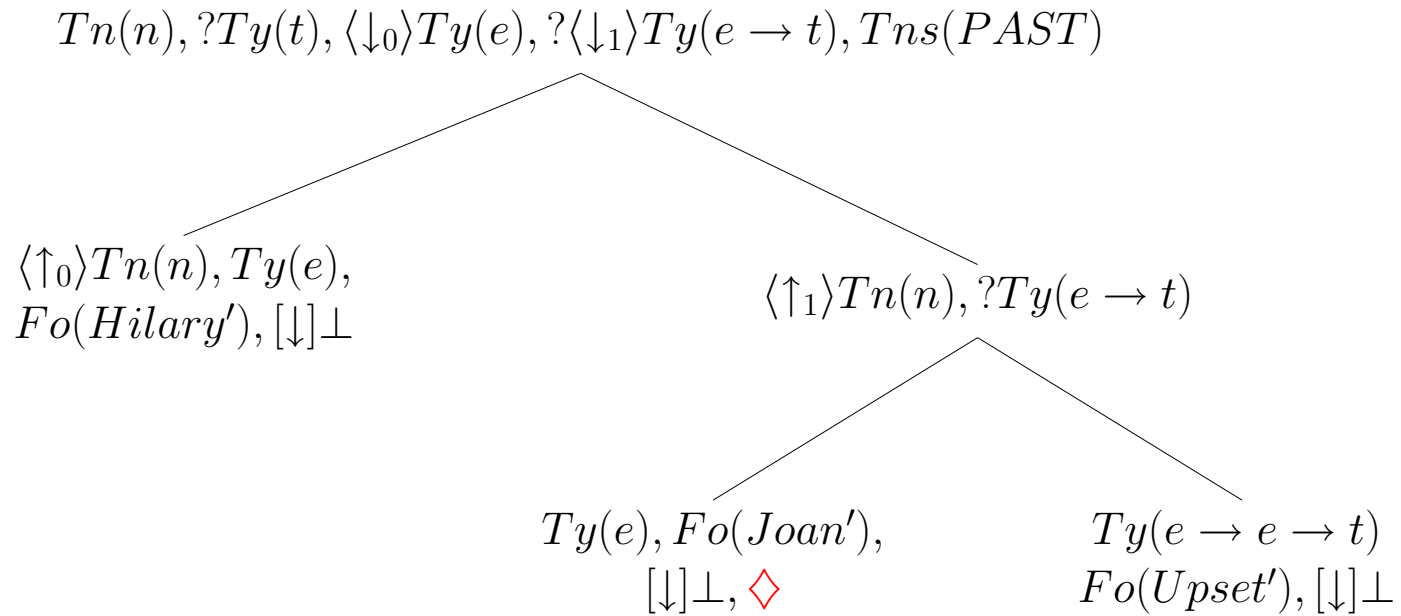
Anticipation

Parsing *Hilary upset Joan*



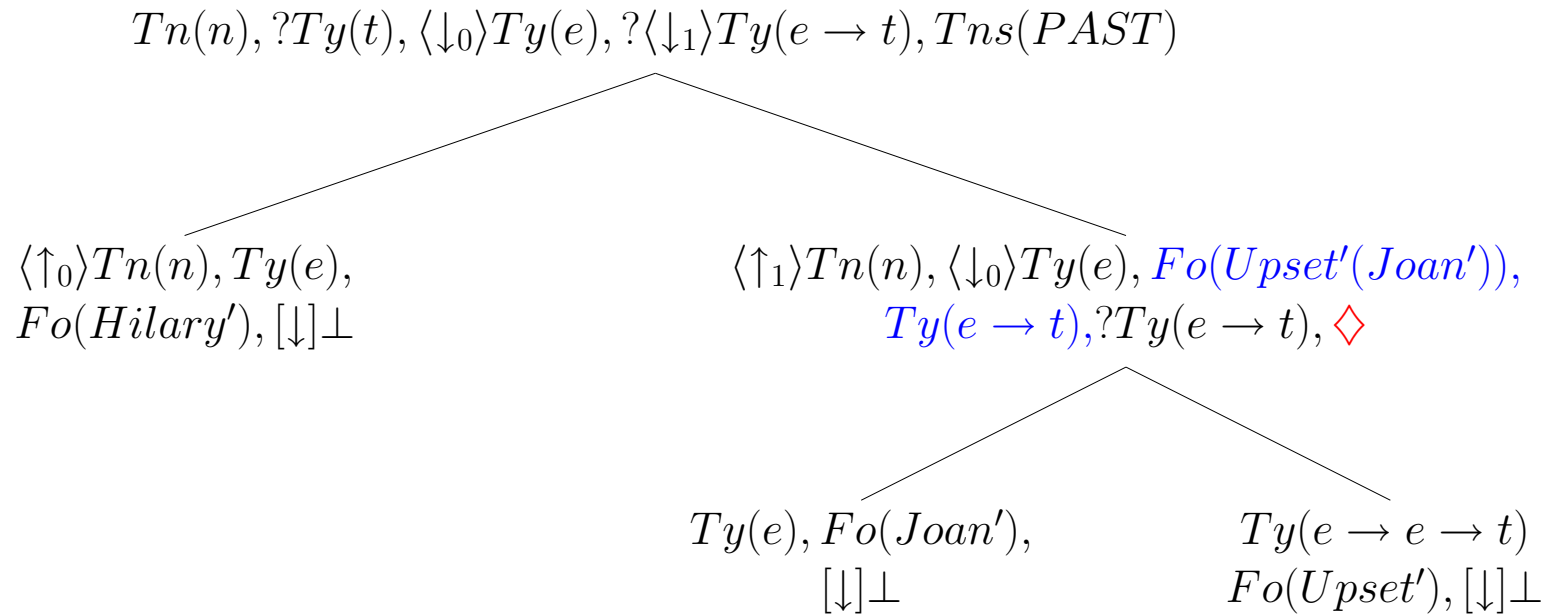
Joan

Parsing *Hilary upset Joan*



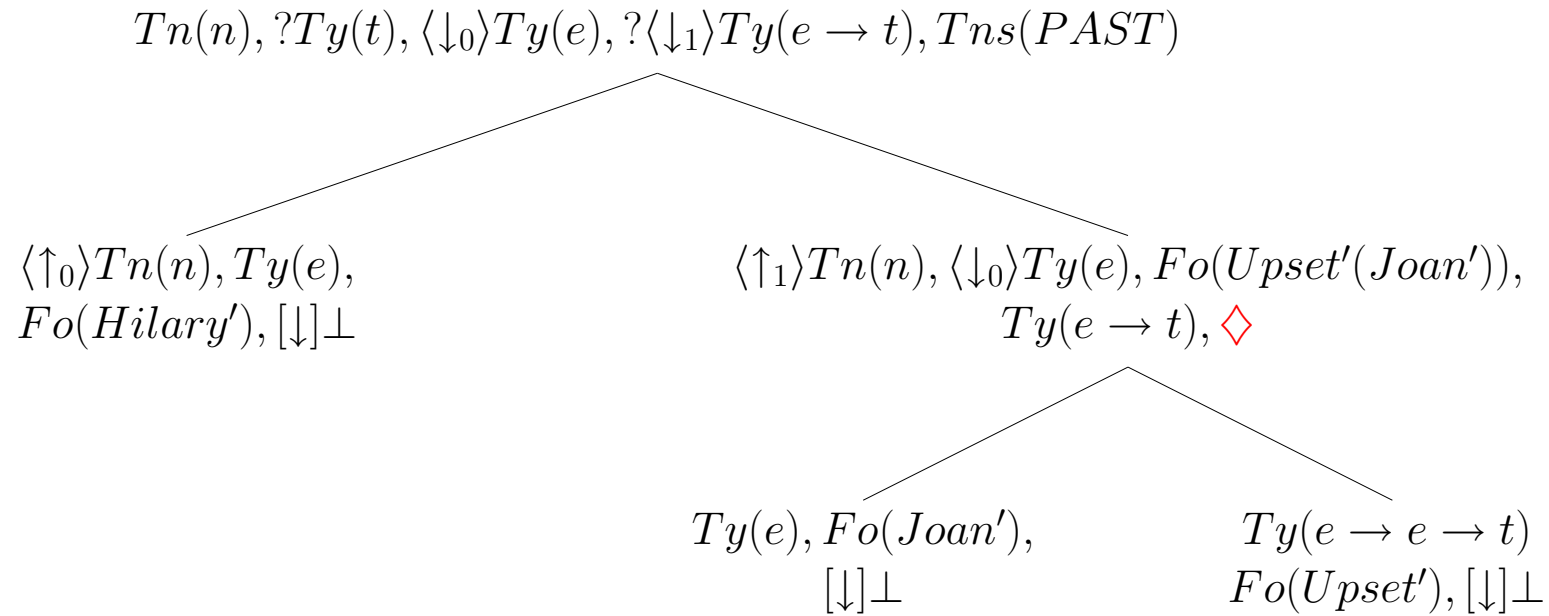
Thinning

Parsing Hilary upset Joan



Elimination

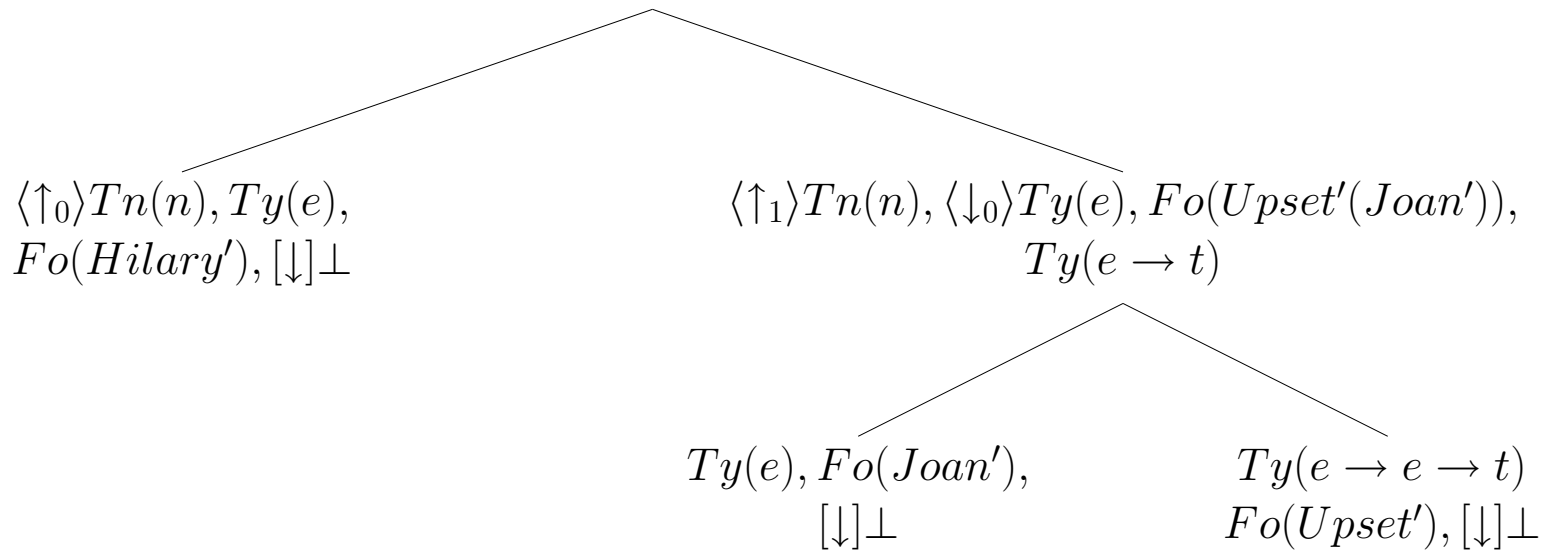
Parsing Hilary upset Joan



Thinning

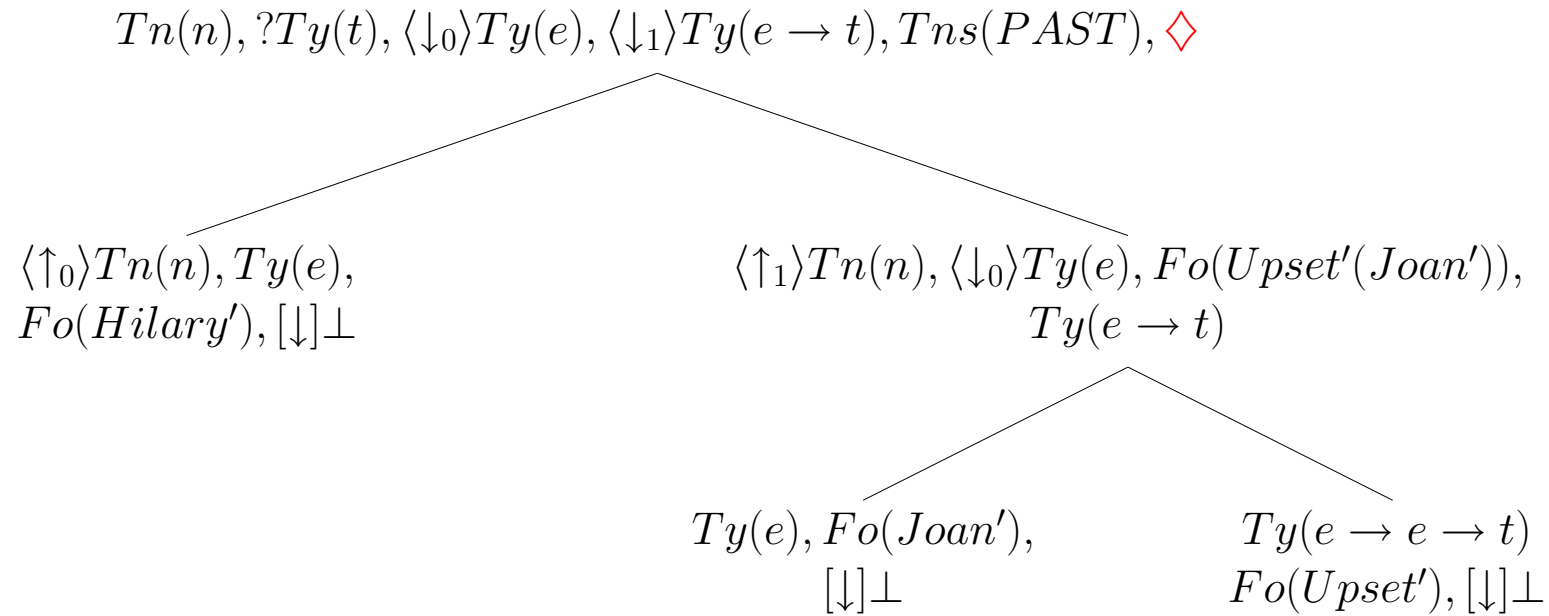
Parsing Hilary upset Joan

$Tn(n), ?Ty(t), \langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \langle \downarrow_1 \rangle Ty(e \rightarrow t), , Tns(PAST), \diamond$



Completion

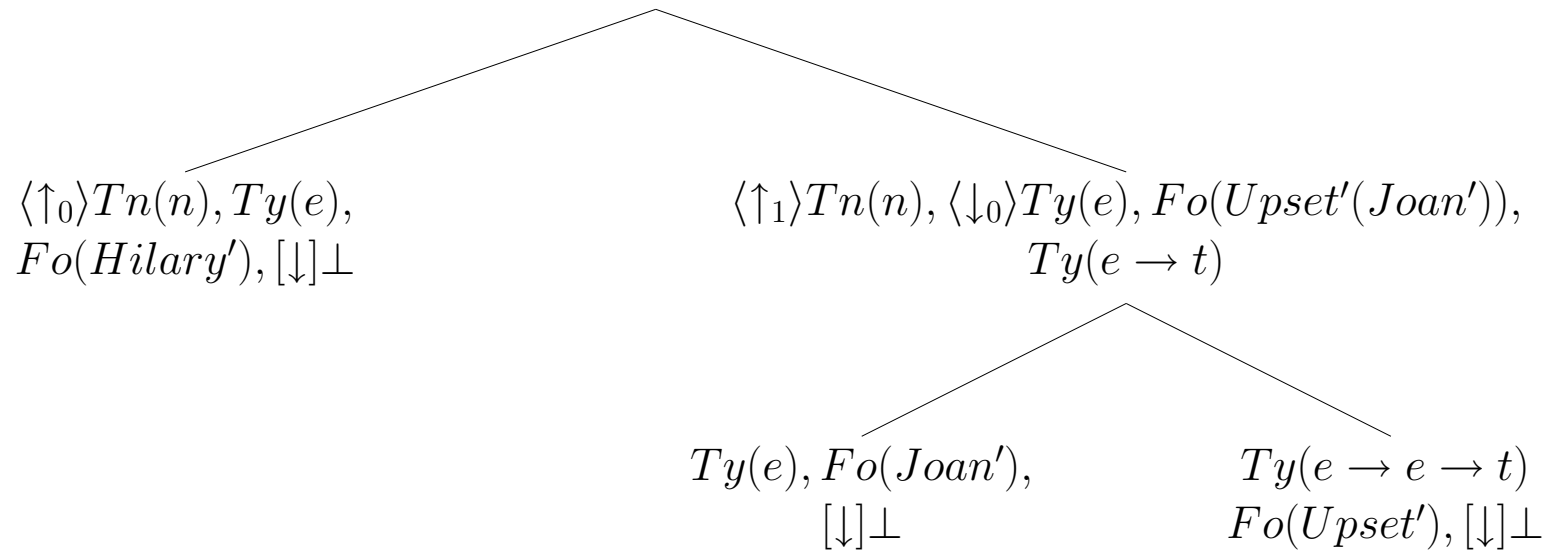
Parsing Hilary upset Joan



Thinning

Parsing Hilary upset Joan

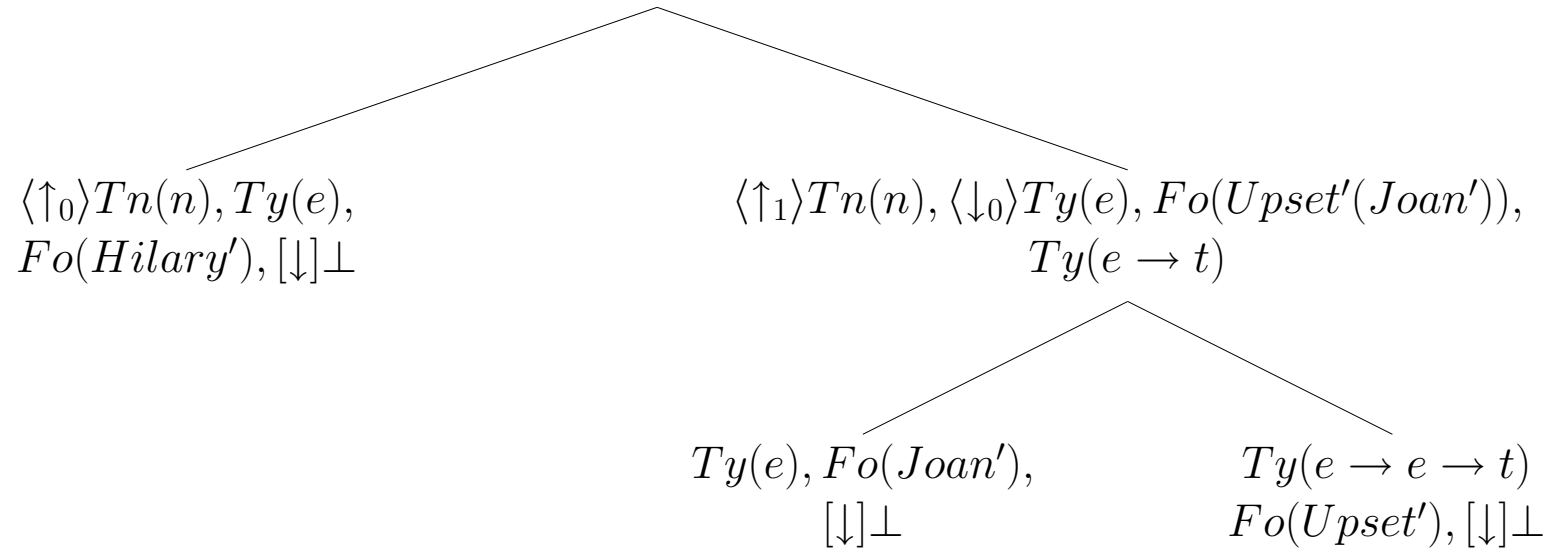
$Tn(n), ?Ty(t), Ty(t), \langle \downarrow_0 \rangle Ty(e), \langle \downarrow_1 \rangle Ty(e \rightarrow t), Tns(PAST), Fo((Upset'(Joan'))(Hilary')), \diamond$



Elimination

Parsing Hilary upset Joan

$Tn(n), Ty(t), \langle \downarrow_0 \rangle Ty(e), \langle \downarrow_1 \rangle Ty(e \rightarrow t), Tns(PAST), Fo((Upset'(Joan'))(Hilary')), \diamond$



Thinning

5 Left Dislocation Structures

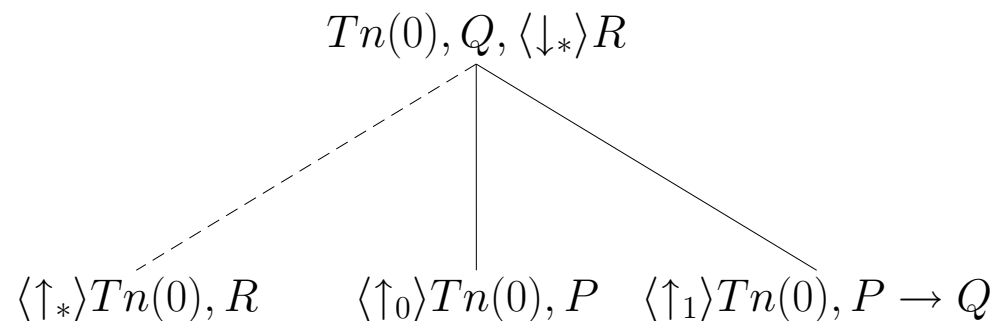
(34) Underspecified modal relations:

- a. $\langle \uparrow_* \rangle$ points to some node that dominates the current node.
- b. $\langle \downarrow_* \rangle$ points to some node dominated by the current node.

(35) $\langle \uparrow_* \rangle \alpha =_{def} \alpha \vee \langle \uparrow \rangle \langle \uparrow_* \rangle \alpha$

(36) $\langle \downarrow_* \rangle \alpha =_{def} \alpha \vee \langle \downarrow \rangle \langle \downarrow_* \rangle \alpha$

(37) An Unfixed Node



*ADJUNCTION

(38) *ADJUNCTION defines a transition:

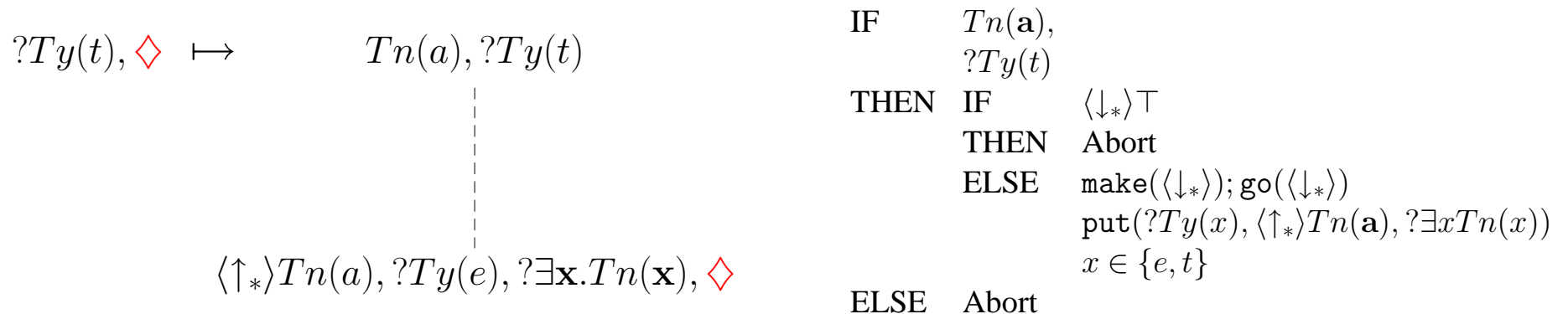
- *from* a partial tree containing only one node with requirement $?Ty(t)$
- *to* one that has an additional node with a type e decoration dominated by the input node and a requirement to find a fixed position within the unfolding tree.

(39) *ADJUNCTION

Rule:

$$\frac{\{\{Tn(a), \dots ?Ty(t), \diamond\}\}}{\{\{Tn(a), \dots, ?Ty(t)\}, \{\langle \uparrow_* \rangle Tn(a), ?\exists x.Tn(x), ?Ty(e), \diamond\}\}}$$

Tree growth:



Left Dislocation

$Tn(n), ?Ty(t), \diamond$

AXIOM

(40) Parsing *Joan, Hilary upset*

Left Dislocation

$$Tn(0), ?Ty(t)$$

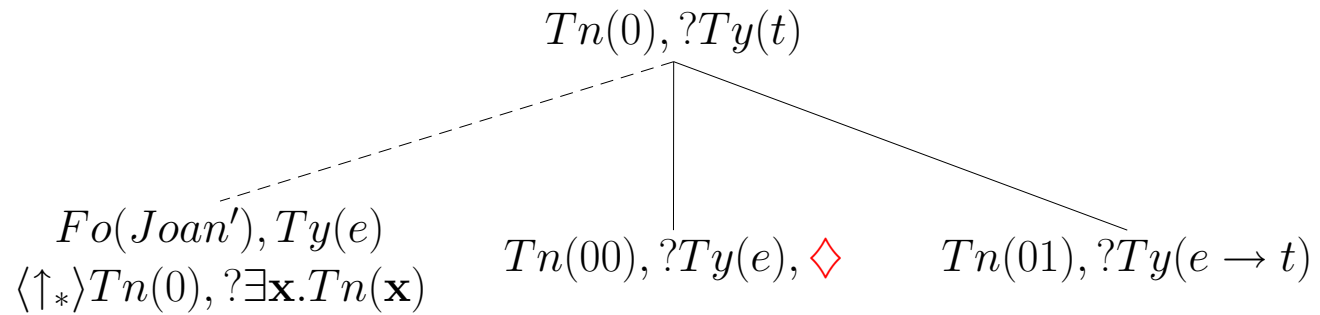
⋮

$$\langle \uparrow_* \rangle Tn(0), Ty(e), Fo(Joan'), ?\exists \mathbf{x}. Tn(\mathbf{x}), \diamond$$

***ADJUNCTION**+Parsing *Joan*

(40) Parsing *Joan*, *Hilary upset*

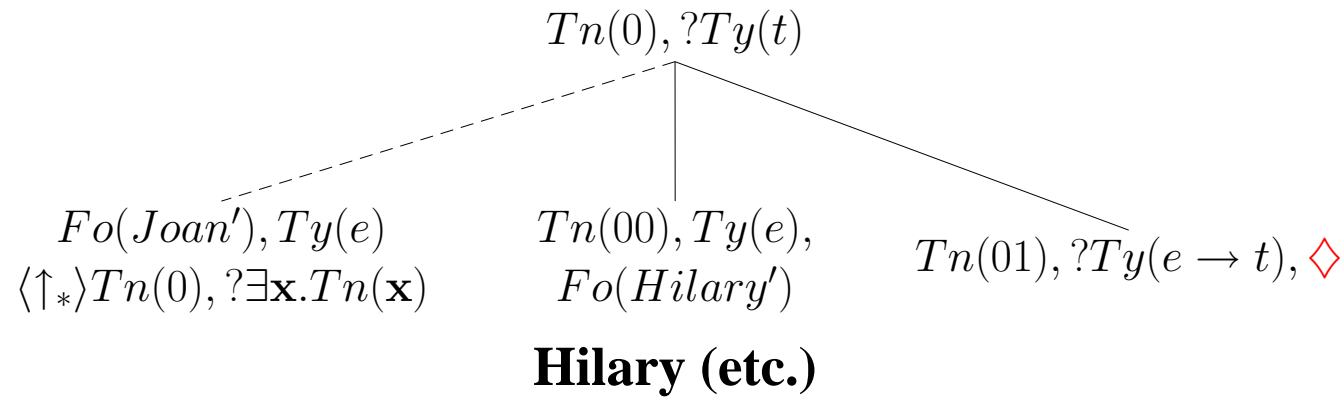
Left Dislocation



INTRODUCTION/PREDICTION

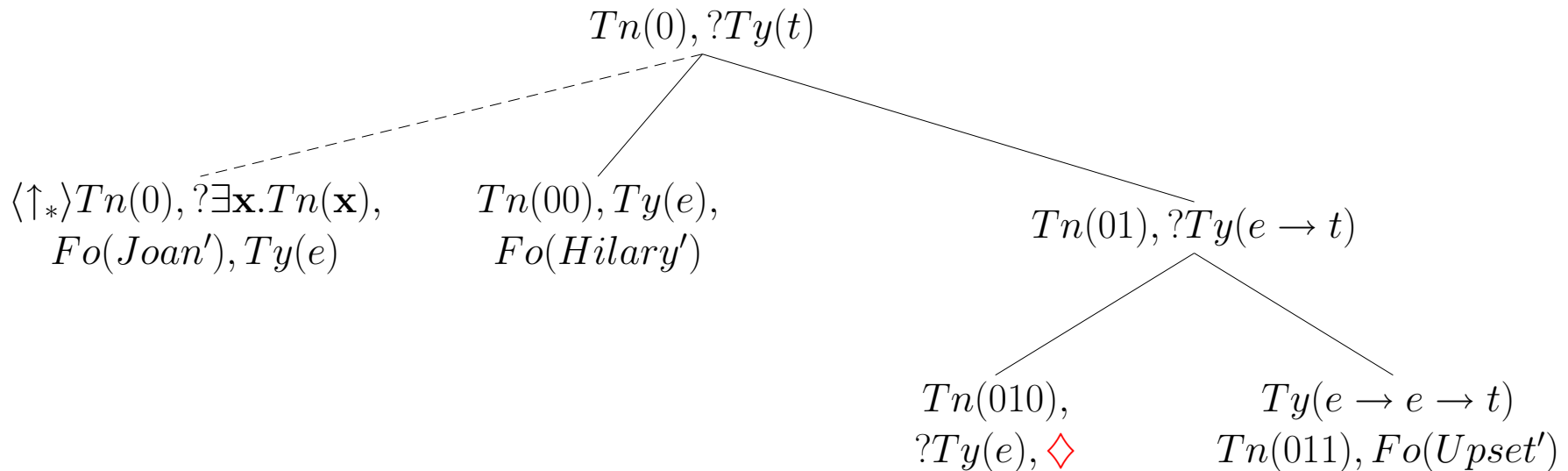
(40) Parsing *Joan, Hilary upset*

Left Dislocation



(40) Parsing *Joan, Hilary upset*

Left Dislocation



Upset

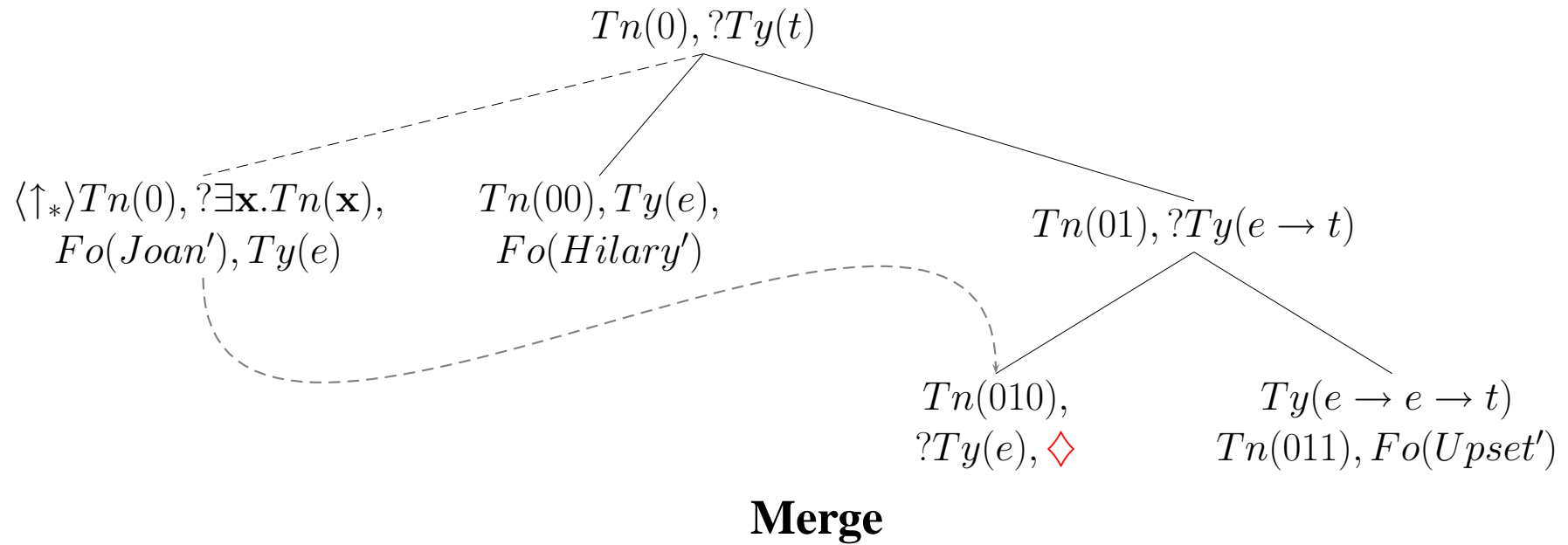
(40) Parsing *Joan, Hilary upset*

(41) MERGE

$$\frac{\{ \dots \mathbf{ND}, \mathbf{ND}' \dots \}}{\{ \dots \mathbf{ND} \sqcup \mathbf{ND}' \dots \}}$$

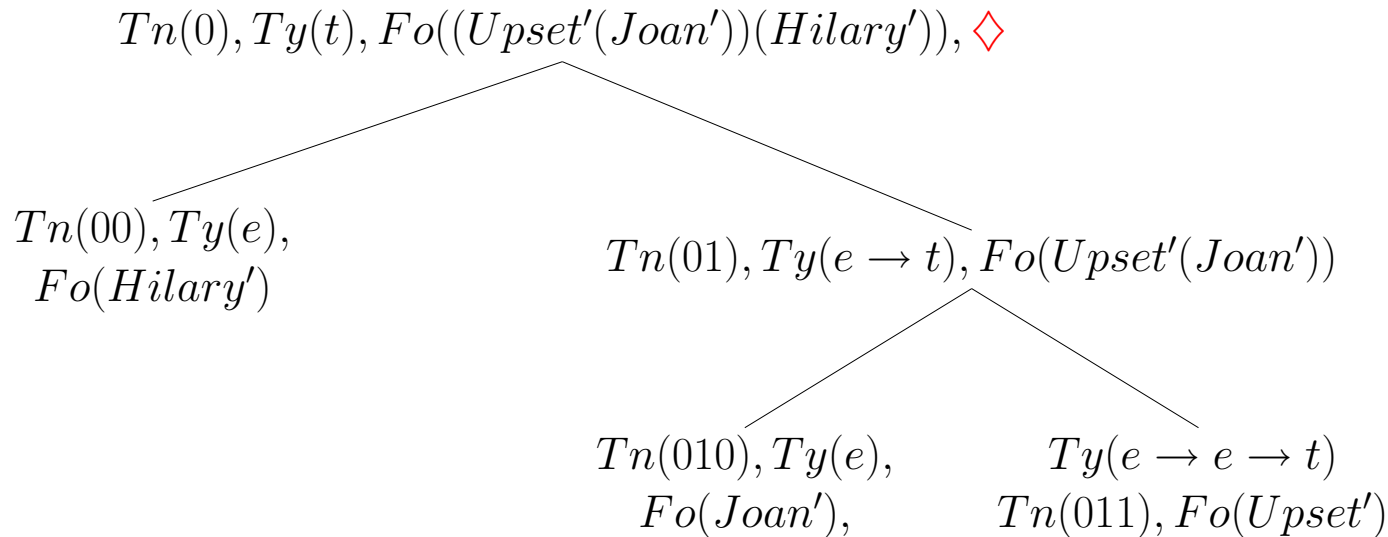
$\diamond \in \mathbf{ND}'$, $\{ \mathbf{ND}, \mathbf{ND}' \}$ are node descriptions

Left Dislocation



(40) Parsing *Joan, Hilary upset* with Merge

Left Dislocation



Completing the tree

(40) Parsing *Joan, Hilary upset*

(43) How the requirements are satisfied:

- a. $\{ \langle \uparrow_* \rangle Tn(0), \langle \uparrow_0 \rangle \langle \uparrow_1 \rangle Tn(0), ?\exists \mathbf{x}. \mathbf{Tn}(\mathbf{x}), \mathbf{Tn}(010), ?Ty(e), Ty(e), Fo(Joan'), [\downarrow] \perp, \diamond \}$
- b. $\{ \langle \uparrow_0 \rangle \langle \uparrow_1 \rangle Tn(0), Tn(010), Ty(e), Fo(Joan'), [\downarrow] \perp, \diamond \}$

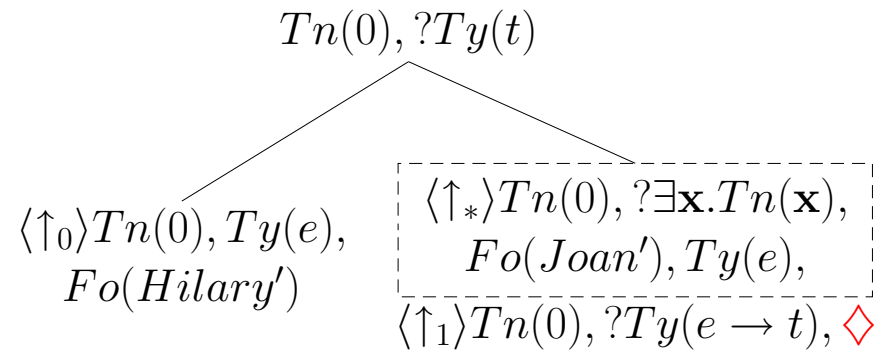
Left Dislocation

$$\{Tn(0), ?Ty(t), \boxed{\langle \uparrow_* \rangle Tn(0), Ty(e), Fo(Joan'), ?\exists \mathbf{x}. Tn(\mathbf{x})}, \diamond\}$$

*ADJUNCTION

(40) Parsing *Joan, Hilary upset*

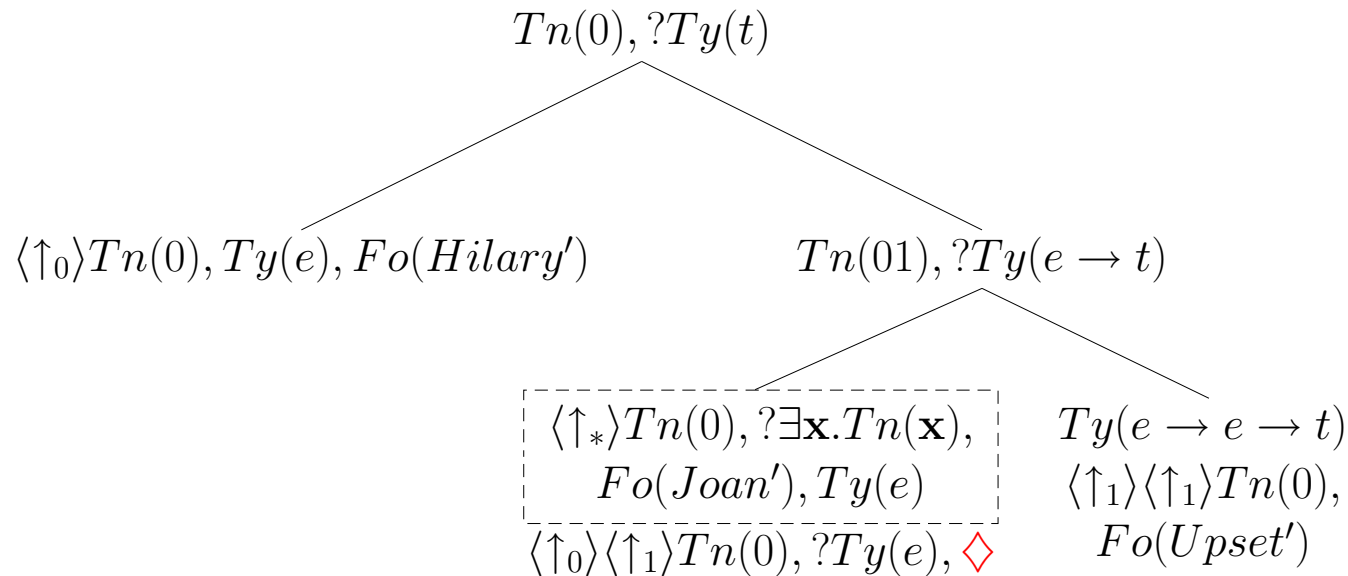
Left Dislocation



Hilary (etc.)

(40) Parsing *Joan, Hilary upset*

Left Dislocation



Upset

(40) Parsing *Joan, Hilary upset*

- Tree then completable as before to yield same tree
- Result: More than one strategy possible at any point in the derivation: alternative derivations always available

Defining locality for resolving dependencies

- (45) Different locality restrictions are definable using different modalities:
- $?\langle \downarrow_* \rangle X$ X must annotate a node within the current tree
 - $?\langle D \rangle X$ X must annotate a node within the current tree or within some linked tree
 - $?\langle \uparrow_0 \rangle \langle \uparrow_*^1 \rangle \langle \downarrow_0 \rangle X$ X must annotate some higher co-argument node
 - $?\langle \uparrow_0 \rangle \langle \downarrow_*^1 \rangle \langle \downarrow_0 \rangle X$ X must annotate some lower co-argument node
 - $?\langle \downarrow_0 \rangle X$ X must annotate my argument daughter node

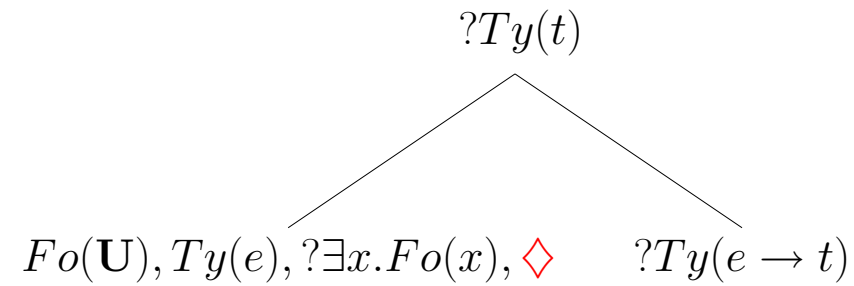
6 Anaphora

- (46) a. Every child thinks that he should get a prize.
 b. Every child thinks that every child should get a prize.
- (47) A unitary characterisation of anaphora?
- | | |
|--|------------------|
| a. Every child thinks that he should get a prize. | bound variable |
| b. They got prizes. | indexical |
| c. Few students enjoyed themselves. They were exhausted. | E-type |
| d. John entered the room. He was smoking. | inter-sentential |
- (48) *Formula* values may have **placeholders** for values: METAVARIABLES in the logical language and represent as boldface capitals **U, V,**
- (49) John ignored Mary. He upset her.

6 Anaphora

(50) Stages in processing *He upset her*

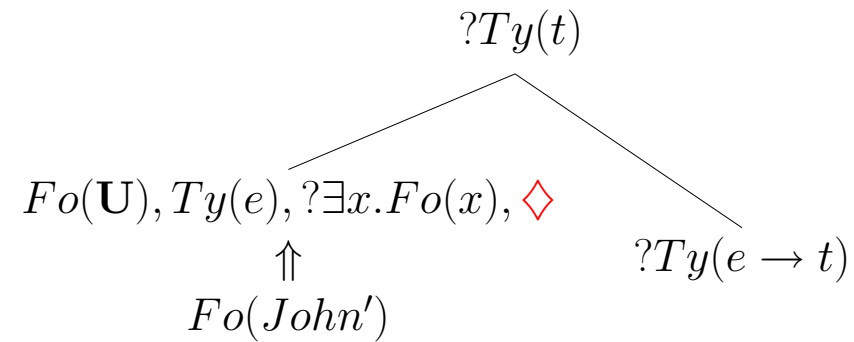
a. Parsing *He*



6 Anaphora

(50) Stages in processing *He upset her*

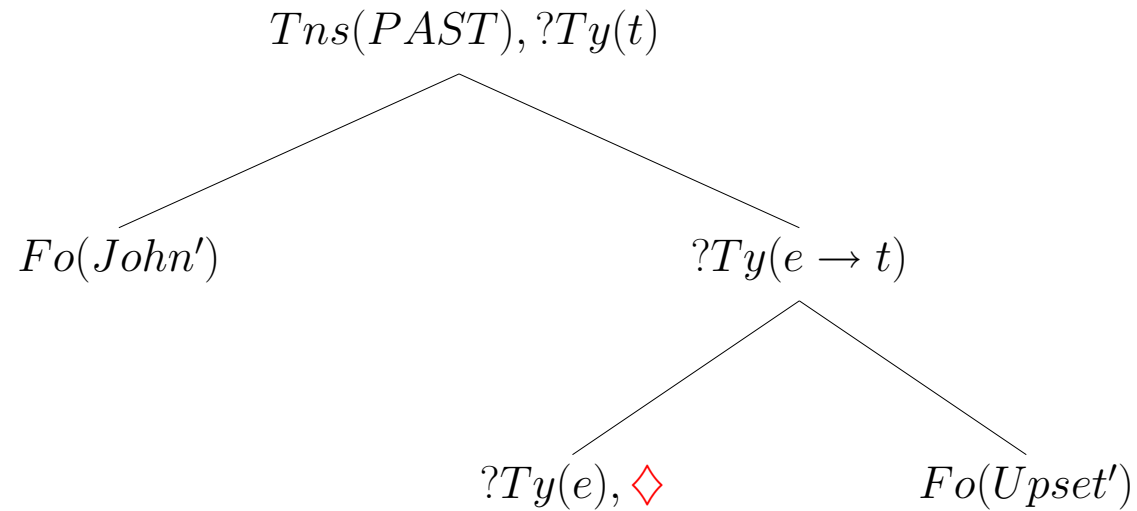
a. Substitution



Anaphora

(50) Stages in processing *He upset her*

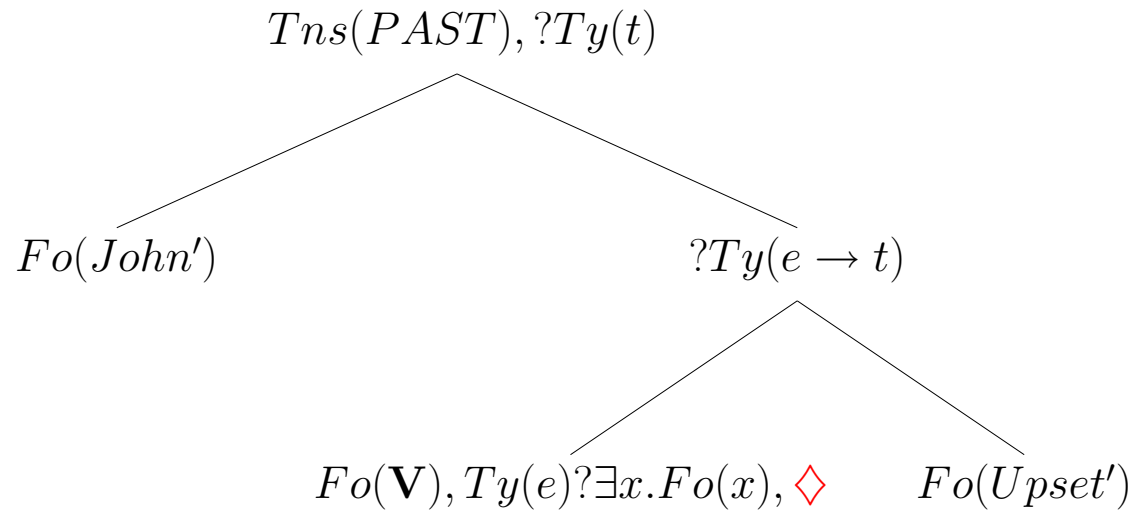
b. Parsing *He upset*



Anaphora

(50) Stages in processing *He upset her*

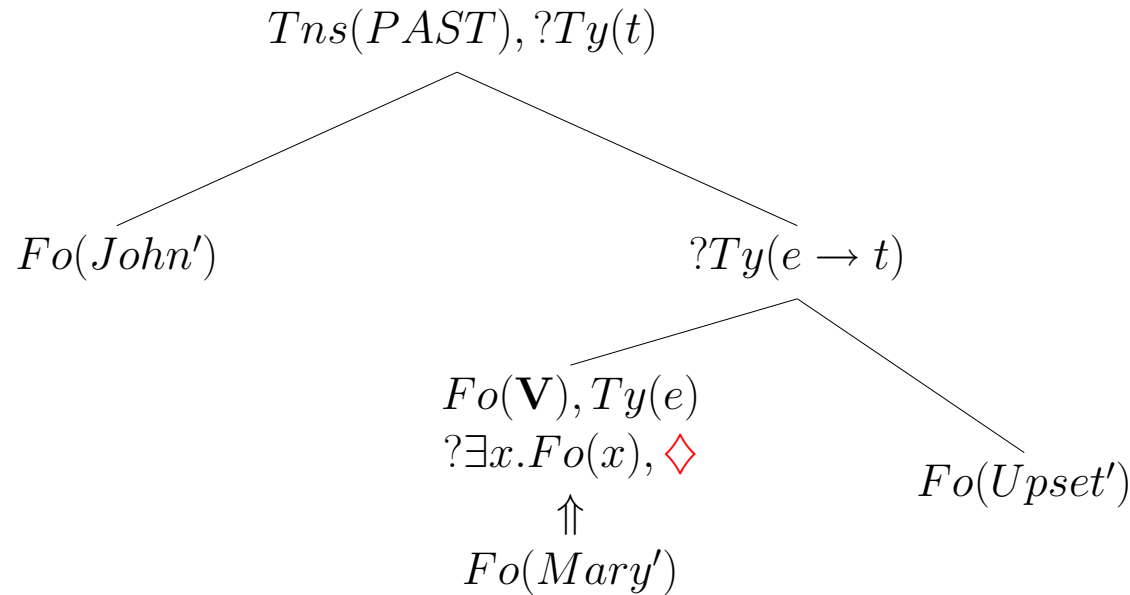
c. Parsing *He upset her*



6 Anaphora

(50) Stages in processing *He upset her*

c. Substitution



| | | | |
|----------------|------------------------|--|---------------------------------|
| (51) <i>he</i> | IF | $?Ty(e)$ | |
| | THEN | put($Ty(e)$, | Trigger |
| | | $Fo(\mathbf{U}_{Male'})$, | Type statement |
| | | $? \exists x.Fo(x)$, | Metavariable and Presupposition |
| | | $? \langle \uparrow_0 \rangle (Ty(t) \wedge \exists \mathbf{x}.Tns(\mathbf{x}))$, | Formula Requirement |
| | $\downarrow [\perp]$) | Case Condition | |
| ELSE | ABORT | Bottom Restriction | |

6 Anaphora

(54) *Subst*(α)

| | |
|---------|---|
| IF | $Fo(\mathbf{U}), Ty(e)$ |
| THEN IF | $\langle \uparrow_0 \rangle \langle \uparrow_1^* \rangle \langle \downarrow_0 \rangle Fo(\alpha)$ |
| | THEN Abort |
| | ELSE $put(Fo(\alpha))$ |
| ELSE | Abort |

(55) *herself*

| | |
|---------|---|
| IF | $?Ty(e)$ |
| THEN IF | $\langle \uparrow_0 \rangle ?Ty(t)$ |
| | THEN Abort |
| | ELSE IF $\langle \uparrow_0 \rangle \langle \uparrow_1^* \rangle \langle \downarrow_0 \rangle Fo(\alpha)$ |
| | THEN $put(Ty(e), Fo(\alpha), [\downarrow] \perp)$ |
| | ELSE Abort |
| ELSE | Abort |

7 Pro drop and Cross-language variation

(56) English verbs

- Trigger is $?Ty(e \rightarrow t)$
- Lexical actions build all *internal* argument structure (i.e. excluding the subject) with pointer at one of the internal argument nodes (variation with ditransitives leads to different patterns, double object and PP object).
- Subject/predicate built by Introduction/prediction
- All arguments obligatory.

(57) Latin verbs

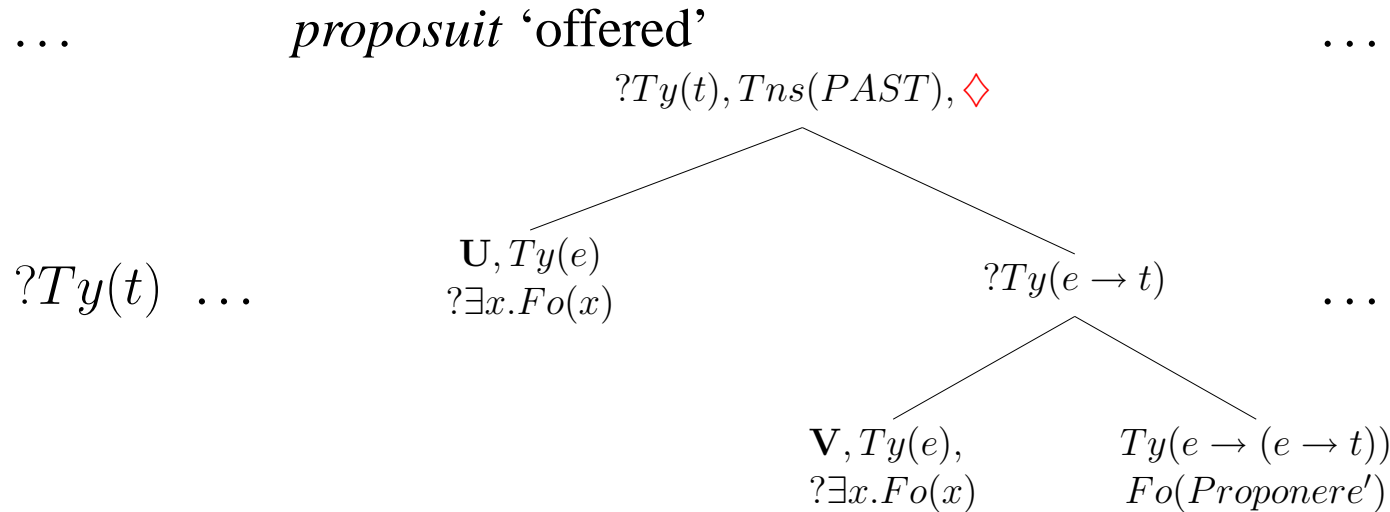
- Trigger is $?Ty(t)$ with no obligatory daughters.
- Lexical actions build all argument structure including the subject with pointer left at top node.
- Consequence: All arguments are optional.

(58) Spanish verbs

- Trigger is $?Ty(t)$ with no obligatory daughters.
- Lexical actions build all argument structure but only the subject has its argument node filled.
- Consequence: subject arguments are optional.

7 Pro drop and optionality

(59i) Verbs induce propositional structure for a predicate – Latin



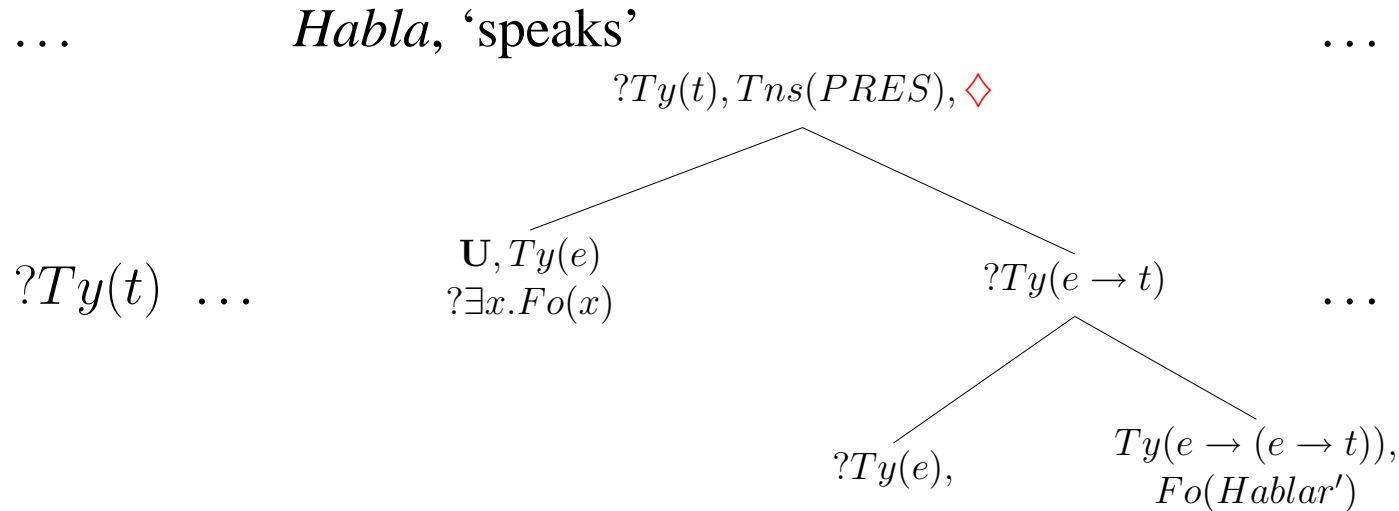
| | | | |
|------|--|--|-------------------|
| IF | $?Ty(t)$ | | |
| THEN | put($Tns(PAST)$); | | Tense |
| | make($\langle \downarrow_1 \rangle$); go($\langle \downarrow_1 \rangle$); put($?Ty(e \rightarrow t)$); | | Predicate Node |
| | make($\langle \downarrow_1 \rangle$); go($\langle \downarrow_1 \rangle$); | | |
| | put($Fo(Proponere\prime), Ty(e \rightarrow e \rightarrow t)$), \perp | | Main Functor |
| | go($\langle \uparrow_1 \rangle$); make($\langle \downarrow_0 \rangle$); go($\langle \downarrow_0 \rangle$); | | |
| | put($Fo(V), Ty(e), ?\exists x.Fo(x)$) | | Internal Argument |
| | go($\langle \uparrow_0 \rangle$); go($\langle \uparrow_1 \rangle$); make($\langle \downarrow_0 \rangle$) : go($\langle \downarrow_0 \rangle$); | | |
| | put($Ty(e), Fo(U), ?\exists x.Fo(x)$); go($\langle \uparrow_0 \rangle$) | | Subject |
| ELSE | Abort | | |

U, V are metavariables requiring a fixed value

(No bottom restriction to argument nodes)

7 Pro drop and optionality

(59ii) Verbs induce propositional structure for a predicate – Spanish



| | | | |
|------|---|---|--|
| IF | $?Ty(t)$ | | |
| THEN | $put(Tns(PRES));$ $make(\langle \downarrow_1 \rangle); go(\langle \downarrow_1 \rangle); put(?Ty(e \rightarrow t));$ $make(\langle \downarrow_1 \rangle); go(\langle \downarrow_1 \rangle);$ $put(Fo(Hablar'), Ty(e \rightarrow e \rightarrow t)), \perp$ $go(\langle \uparrow_1 \rangle); make(\langle \downarrow_0 \rangle); go(\langle \downarrow_0 \rangle);$ $put(?Ty(e),$ $go(\langle \uparrow_0 \rangle); go(\langle \uparrow_1 \rangle); make(\langle \downarrow_0 \rangle) : go(\langle \downarrow_0 \rangle);$ $put(Ty(e), Fo(U), ?\exists x.Fo(x)); go(\langle \uparrow_0 \rangle)$ | Tense Predicate Node Main Functor Internal Argument Subject | U, V are metavariables requiring a fixed value |
| ELSE | Abort | | |

(No bottom restriction to argument nodes))

7 Pro drop, optionality and rule interaction(a)

- (60) processing *Praemium proposuit*
 Reward offered
 (Xerxes) offered a reward.

Steps used in parsing *praemium*

$$Tn(0), ?Ty(t), \diamond$$

$$\langle \uparrow_* \rangle Tn(0), Ty(e), Fo(Praemium'), ?\exists \mathbf{x}. Tn(\mathbf{x})$$

Rules used:

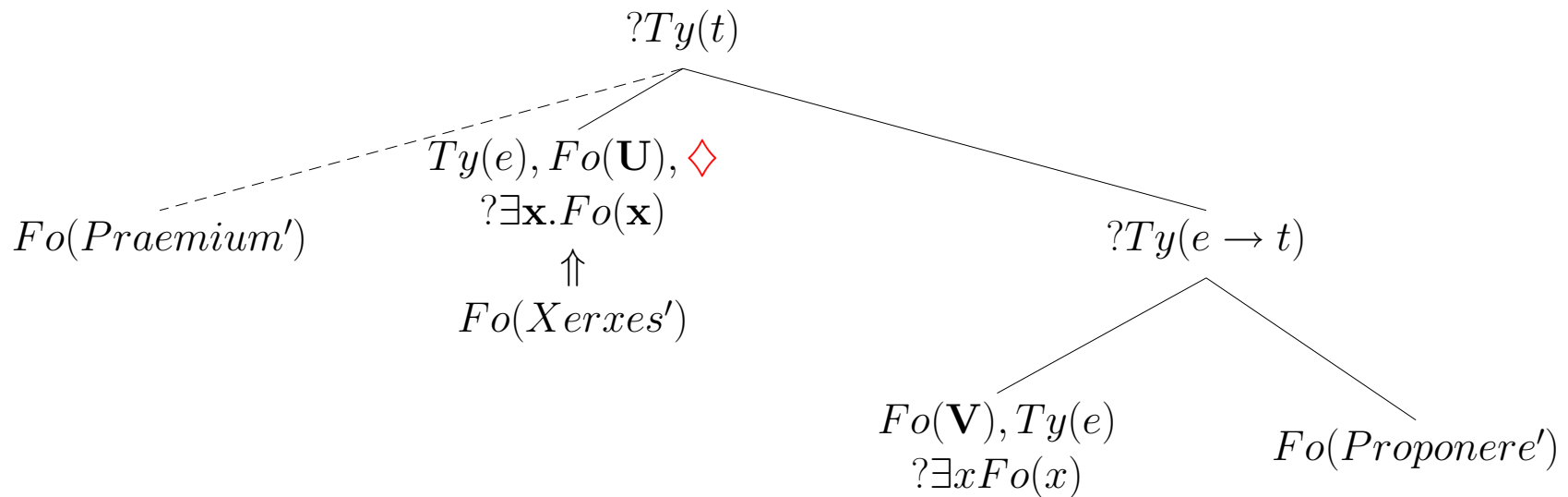
*Adjunction, Lexical scanning, Thinning, Completion

Pro drop, optionality and rule interaction (b)

– Interaction of pragmatics and syntax

(60) processing *Praemium proposuit*
 Reward offered
 (Xerxes) offered a reward.

Parsing the verb and identifying the subject



Rules used:

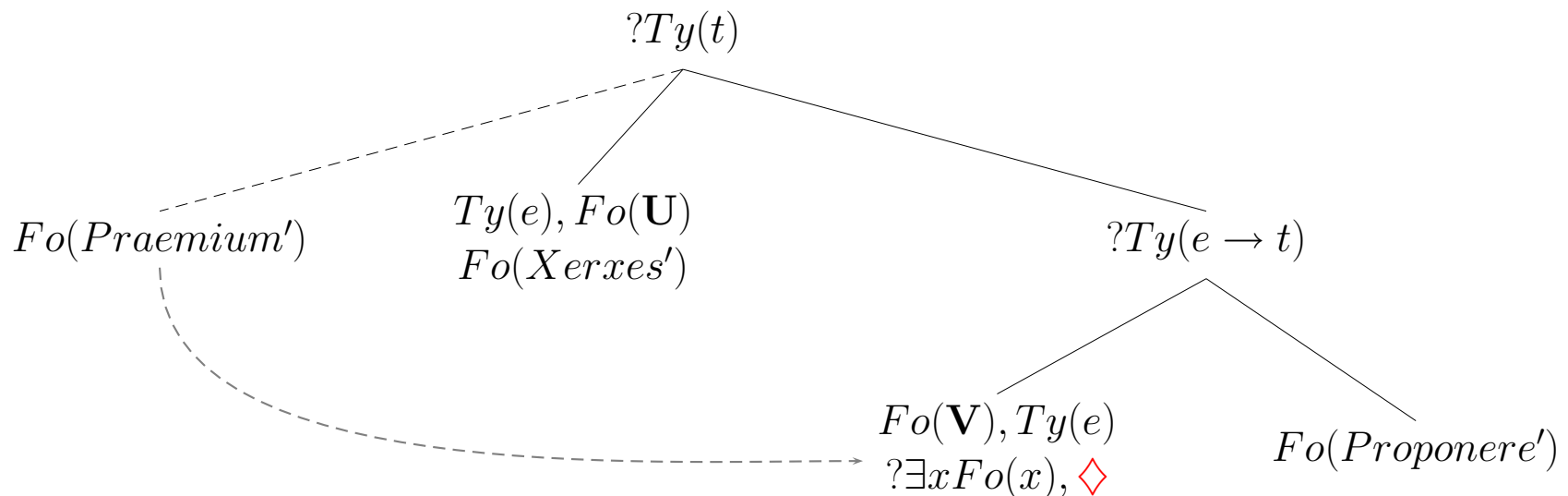
Lexical scanning (of *proposuit*),

Anticipation to get to subject node, Substitution, Thinning

Pro-drop, optionality and order variation (c)

- (60) processing *Praemium proposuit*
 Reward offered
 (Xerxes) offered a reward.

Unifying the unfixed node with the object node



Rules used:

Completion (to get up from subject node),

Anticipation twice to get to object node,

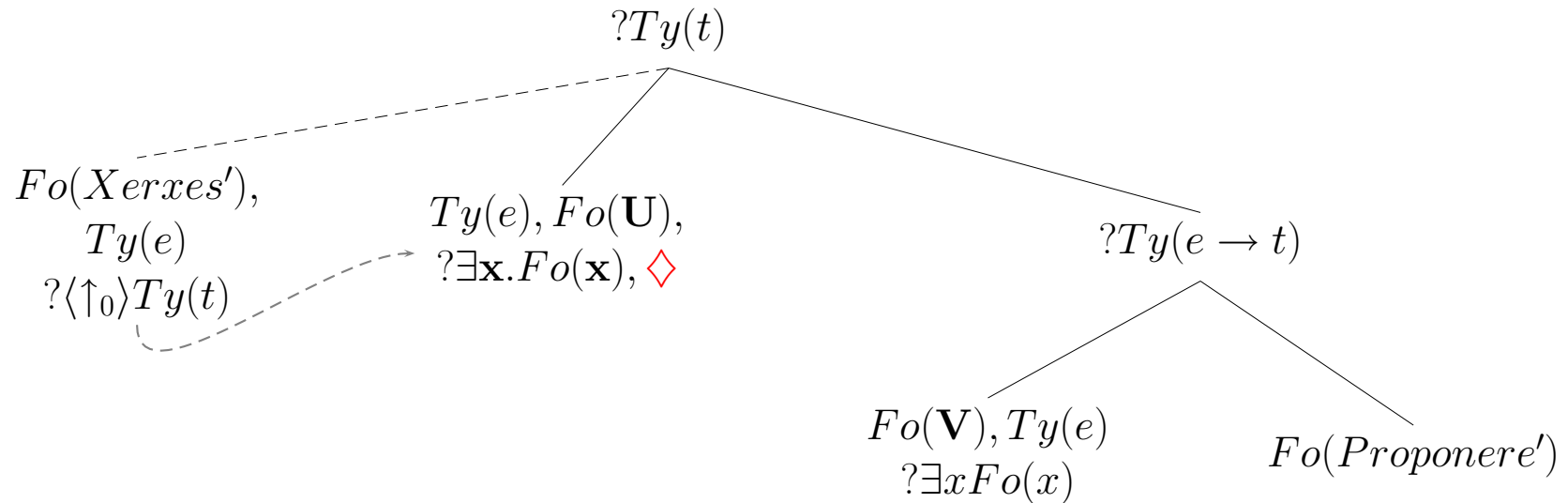
”Merge” unifying the unfixed and object nodes

Completion, Elimination, Thinning to yield fully decorated tree.

pro-drop, optionality and case-specifications

(61) Parsing *Xerxes proposuit* ‘Xerxes_{NOM} offered it’:

Unifying the case-marked unfixed node with the subject node
(case as an output filter):



The requirement $? \langle \uparrow_0 \rangle Ty(t)$ is met at the final step, when a $Ty(t)$ formula is established.

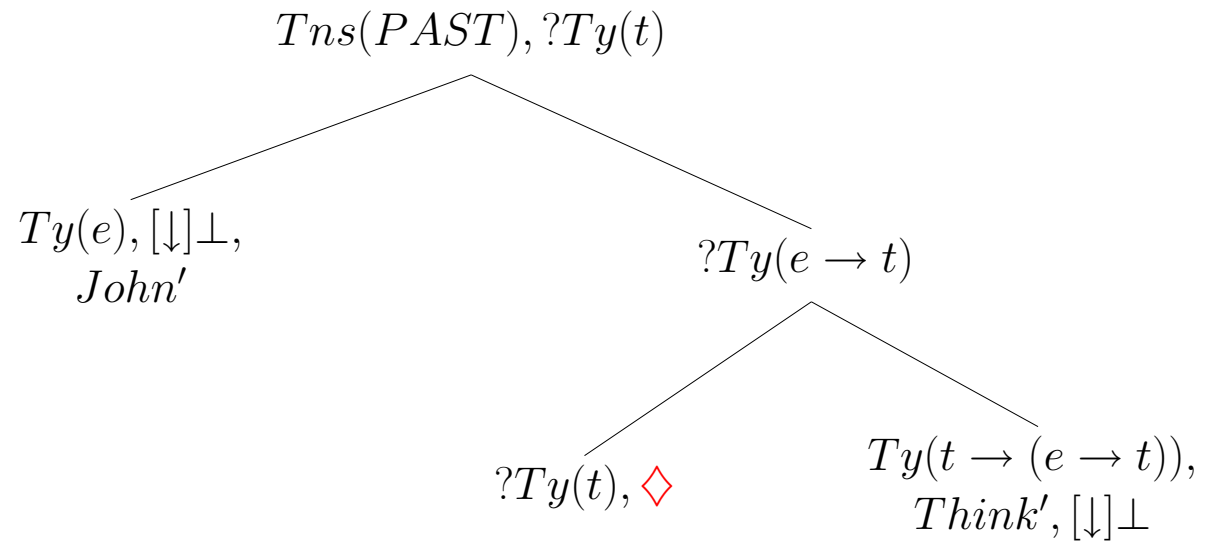
The unification of the unfixed node with the subject node is unproblematic for all NPs, despite the metavariable decorating it. The argument node projected by the verb has no bottom restriction.

Complement Clauses

(62) *thought*

| | |
|------|---|
| IF | $?Ty(e \rightarrow t)$ |
| THEN | $go(\langle \uparrow_1 \rangle); put(Tns(PAST)); go(\langle \downarrow_1 \rangle);$ $make(\langle \downarrow_1 \rangle); go(\langle \downarrow_1 \rangle); put(Ty(t \rightarrow e \rightarrow t), Fo(Think'), [\downarrow] \perp);$ $go(\langle \uparrow_1 \rangle); make(\langle \downarrow_0 \rangle); go(\langle \downarrow_0 \rangle); put(?Ty(t))$ |
| ELSE | ABORT |

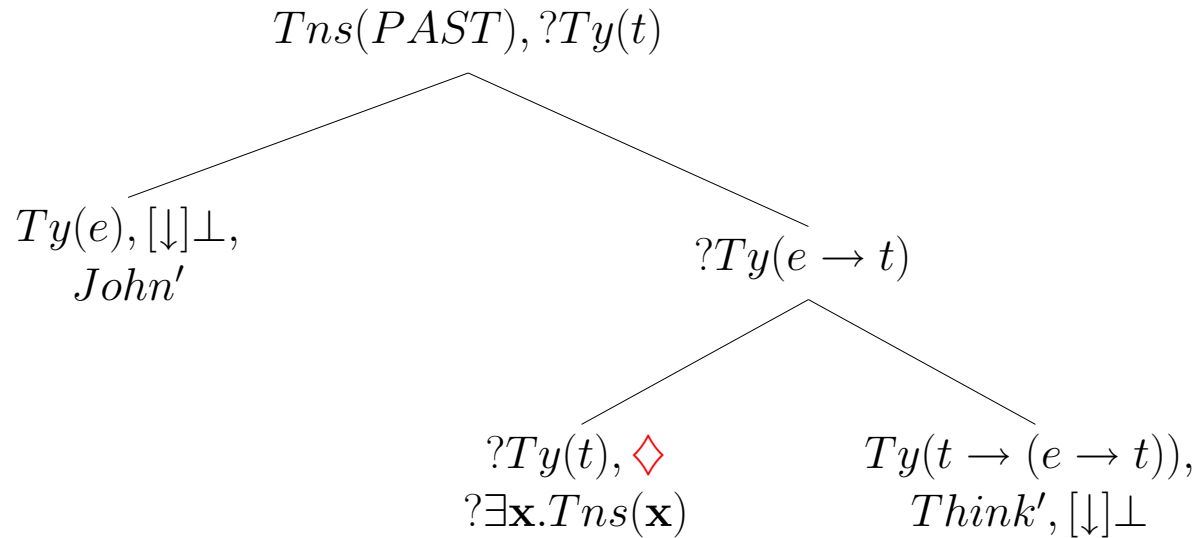
(63) Parsing *John thought*



Complement Clauses

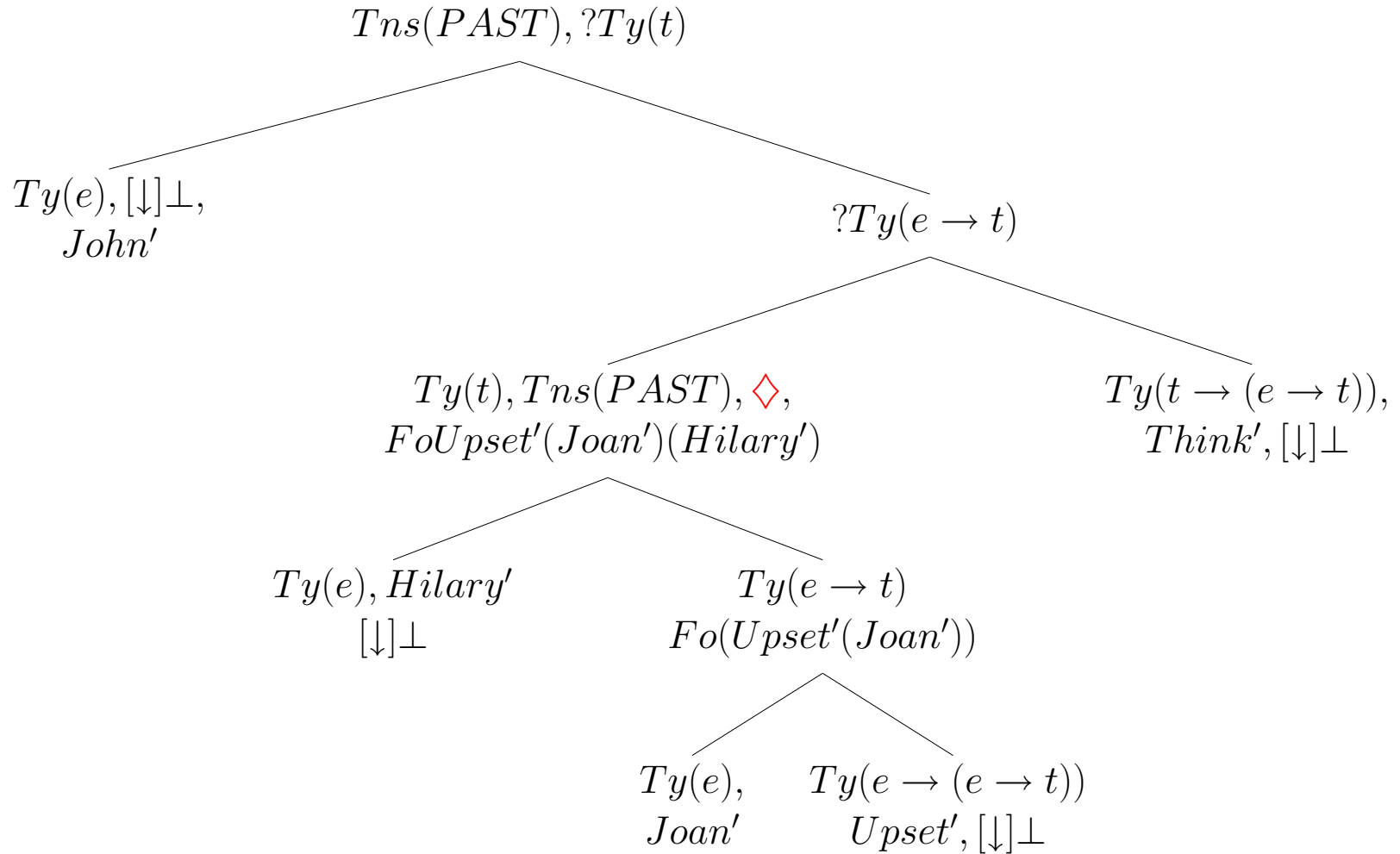
(64) *that*_{comp} $\left\{ \begin{array}{l} \text{IF } ?Ty(t) \\ \text{THEN IF } \langle \uparrow \rangle \perp \\ \text{THEN ABORT} \\ \text{ELSE put}(?\exists \mathbf{x}.Tns(\mathbf{x})) \\ \text{ELSE ABORT} \end{array} \right.$

(65) Parsing *John thought that*



Complement Clauses

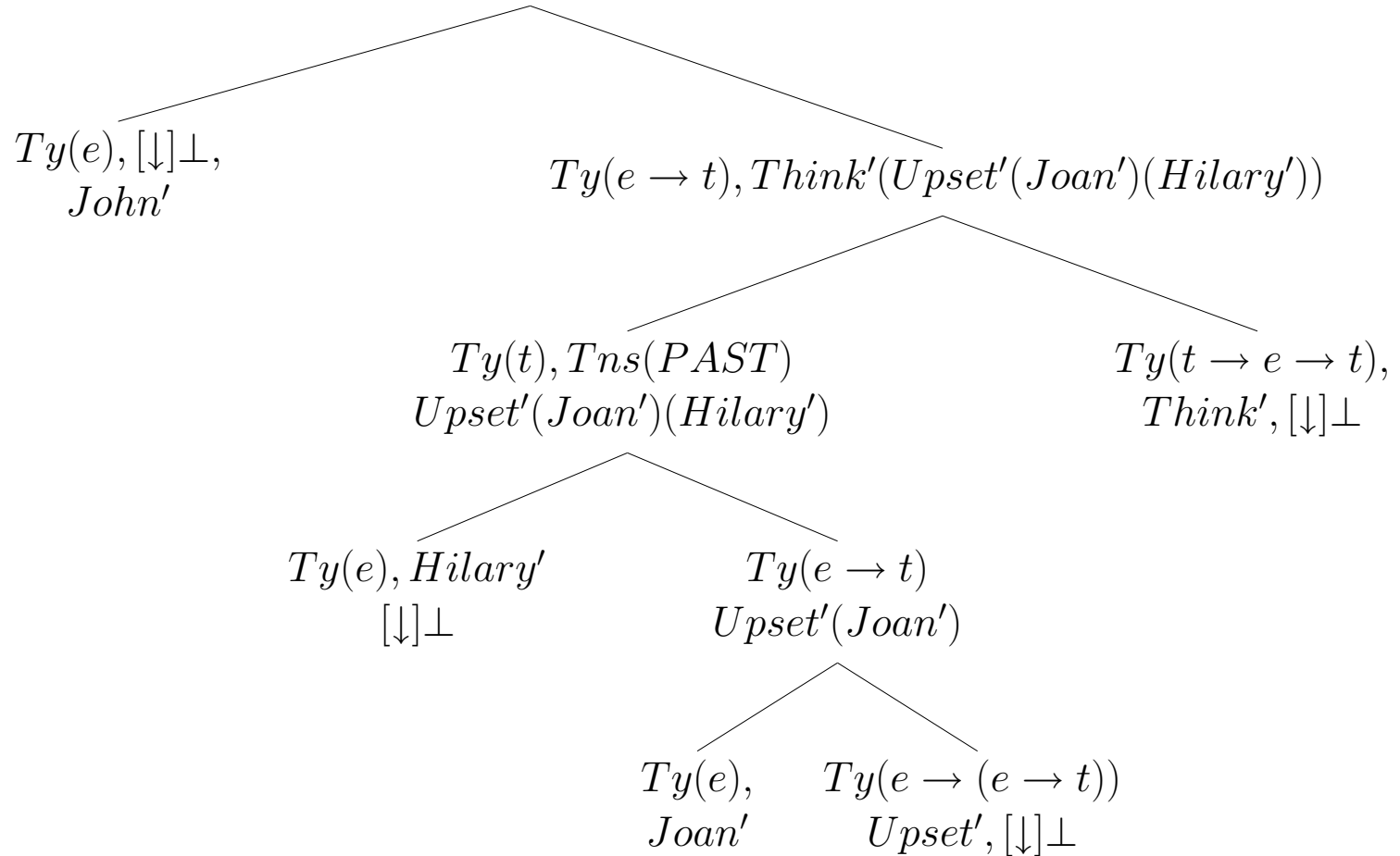
(66) Parsing *John thought that Hilary upset Joan*



Complement Clauses

(67) Completing *John thought that Hilary upset Joan*

$Tns(PAST), Think'(Upset'(Joan')(Hilary'))(John'), Ty(t), \diamond$



Grammaticality and well-formedness

(68) Tree growth:

- * Decorations are invariably added to the tree: nothing is taken away except requirements once fulfilled.
- * Growth is progressive: no trivial additions are allowed.

The process of growth is UPWARD MONOTONIC

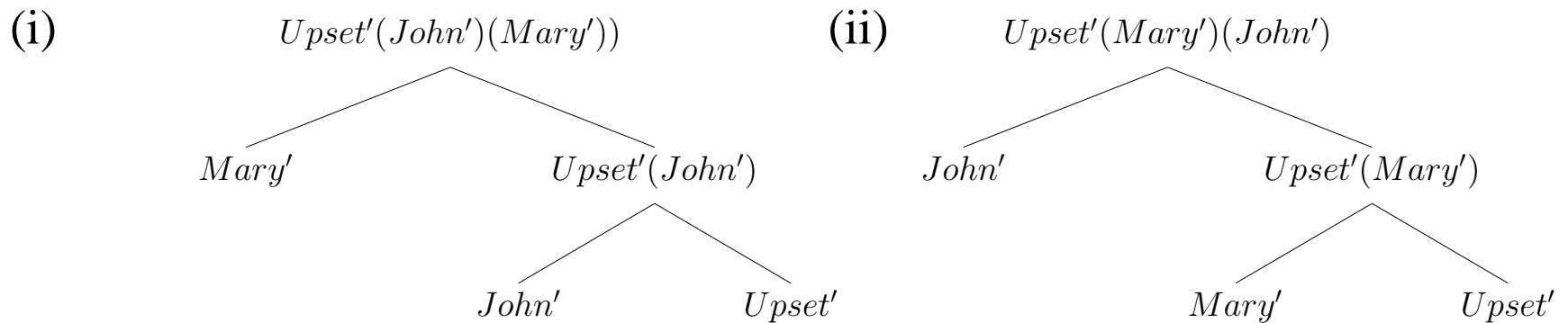
- * Starting point is the axiom, $?Ty(t)$: output for all successful sequences is achieving that goal, with no outstanding requirements.
- * All goals and subgoals have to be met by following the actions as provided by the words in the linear sequence in which the words are presented. No actions from any word can be put on hold, and used at some arbitrary point later.

(69) To be a wellformed string: there must be a sequence of actions that follows from the initial goal to the construction of a logical form decorating the top node of a tree with actions from each word having been used in the sequence presented in combination with computational actions as applicable, with no requirements outstanding in the end result.

Grammaticality and well-formedness

(70) To be ungrammatical: there must be no possible sequence of transitions using the information presented incrementally by the words in combination with computational actions that yield a single logical form.

(71) How is it that the system guarantees that *John, Mary upset* must map onto the tree equivalent to the string *Mary upset John* (i) and not onto that equivalent to the string *John upset Mary* (ii)?

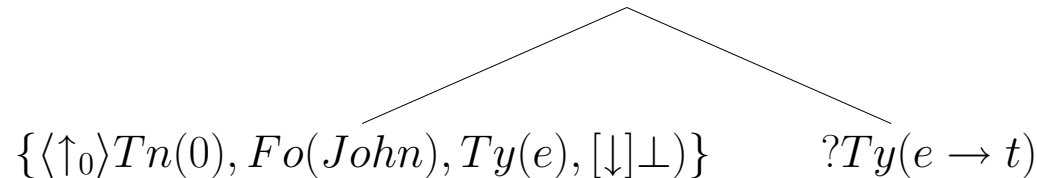


Grammaticality and well-formedness

(72) Two choices in parsing *John, Mary upset*:

- a. *Adjunction: *John* decorates an unfixed node with $Fo(John')$ - leads to a well-formed outcome.
- b. Introduction and Prediction: *John* decorates the subject node - does not lead to a well-formed outcome.

$$\{Tn(0), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond\}$$



Neither $?Ty(t)$ nor $?Ty(e \rightarrow t)$ provide an appropriate trigger for parsing *Mary*.

*Adjunction cannot apply: since

$$\{\{Tn(a), \dots ?Ty(t), \diamond\}\}$$

does not hold of the tree whose description from the relevant node

is: $\{\{Tn(0), ?Ty(t), ?\langle \downarrow_0 \rangle Ty(e), ?\langle \downarrow_1 \rangle Ty(e \rightarrow t), \diamond\}$
 $\{\langle \uparrow_0 \rangle Tn(0), Fo(John'), Ty(e), [↓]⊥\}\{\langle \uparrow_1 \rangle Tn(0), ?Ty(e \rightarrow t)\}\}$

Summary

- Using logical types as syntactic constructs, compositionality of content can be modelled as syntactic tree-building
- With a language for describing trees, tree-growth processes are definable
 - * movement replaced by structural underspecification,
 - * case specifications replaced by growth requirements and update
- With lexical/syntactic/pragmatic specifications in same vocabulary, feeding relations between structural/lexical/pragmatic actions are straightforwardly definable
- With syntax/lexicon as building of semantic representations, optionality of arguments is unproblematic.
- With parsing perspective as background, availability of alternative derivations does not constitute ambiguity of structure
- The grammar formalism sets out space of possible alternatives. It does not dictate choices.

Dynamic Syntax References

- Blackburn, P. and Meyer-Viol, W. 1994. “Linguistics, logic and finite trees”. *Bulletin of the Interest Group for Pure and Applied Logic* 2: 3-29.
- Cann, R., Kempson, R. and Marten, L. 2005. *The Dynamics of Language: An Introduction*. Amsterdam: Elsevier.
- Cann, R., Kempson, R., and Purver, M. 2007. ‘Context-dependent wellformedness: the dynamics of ellipsis’. *Research on Language and Computation* 5, 333–58.
- Gregoromichelaki, E. 2006. ‘Conditionals in Dynamic Syntax’. Unpublished PhD thesis, London University.
- Kempson, R., Cann, R. and Kiaer, J. 2006. “Topic, focus and the structural dynamics of language”. In *The Architecture of Focus*, S. Winkler and V. Molnár (eds.). Berlin, New York: Mouton de Gruyter.
- Kempson, R., Kiaer, J., and Cann, R. 2007. “Periphery effects and the dynamics of tree growth”. In Shaer, B., Frey, W. and Maienborn C. (eds.) *Dislocation: Syntactic, semantic, and pragmatic perspectives*. Berlin: de Gruyter.
- Kempson, R., Meyer-Viol M. and Gabbay D. 2001. *Dynamic Syntax*. Oxford: Blackwell.
- Marten, L. 2002. *At the Interface of Syntax and Pragmatics*. Oxford University Press.
- Marten, , R. Kempson and M. Bouzouita, to appear, Concepts of structural underspecification in Bantu and Romance. In Ccile de Cat and Katherine Demuth, eds., *The Romance-Bantu Connection*, Amsterdam: Benjamins.
- Purver, M., Cann, R., and Kempson, R. 2006. ‘Grammars as parsers: meeting the dialogue challenge’. *Research on Language and Computation* 4, 289–326.