Semantic Theory

Lecture 13: Discourse Semantics I

Manfred Pinkal

FR 4.7 Computational Linguistics and Phonetics

Summer 2014

A Problem with Definite NPs

Standard type-theoretic representation of definite NPs:

the ⇒ $\lambda F \lambda G \exists y[\forall x(F(x) \leftrightarrow x{=}y) \wedge G(y)]$

the sun ⇒ $\lambda G \exists y[\forall x[sun'(x) \leftrightarrow x{=}y] \wedge G(y)]$

the sun is shining ⇒ $\exists y[\forall x[sun'(x) \leftrightarrow x{=}y) \wedge shine'(y)]$

the student is working ⇒ $\exists y[\forall x[stud'(x) \leftrightarrow x{=}y) \wedge work'(y)]$

Context-dependent expressions

- **Deictic expressions** depend on the physical utterance situation:
  - *I, you, now, here, this, …*
- **Anaphoric expressions** refer to the linguistic context / previous discourse:
  - *he, she, it, then, …*

A simple context theory

- Model contexts as vectors: sequences of semantically relevant context data with fixed arity.
- Model meanings as functions from contexts to denotations – more specifically, as functions from specific context components to denotations.

An Example

- Context $c = \langle a, b, l, t, r \rangle$
  - *a* speaker
  - *b* addressee
  - *l* utterance location
  - *t* utterance time
  - *r* referred object

$$[\![I]\!]^{M,g,c} = \text{utt}(c) = a$$

$$[\![you]\!]^{M,g,c} = \text{adr}(c) = b$$

$$[\![here]\!]^{M,g,c} = \text{loc}(c) = l$$

$$[\![now]\!]^{M,g,c} = \text{time}(c) = t$$

$$[\![this]\!]^{M,g,c} = \text{ref}(c) = r$$

Type-theoretic context semantics

- **Model structure:** $M = \langle U, C, V \rangle$
  - U – model universe
  - C – context set
  - V – value assignment function that assigns non-logical constants functions from contexts to denotations of appropriate type.
- **Interpretation:**
  - $[\![\alpha]\!]^{M,h,c} = V(\alpha)(c)$, if α is a non-logical constant
  - $[\![\alpha]\!]^{M,h,c} = h(\alpha)$, if α is a variable
  - $[\![\alpha(\beta)]\!]^{M,h,c} = [\![\alpha]\!]^{M,h,c}([\![\beta]\!]^{M,h,c})$

An example

- *I am reading this book* ⇒ read’(this-book’)(I’)
- $[\![\text{read'(this-book')(I')}]\!]^{M,h,c} = 1$
  - iff $[\![\text{read'}]\!]^{M,h,c}([\![\text{this-book'}]\!]^{M,h,c})([\![\text{I'}]\!]^{M,h,c}) = 1$
  - iff V(read’)(ref(c))(utt(c)) = 1
- Context-invariant expressions are constant functions:
  - V(read’)(c) = V(read’)(c’) for all c, c’ $\in$ C

Context-dependence of definite NPs

- Definite NPs pick an appropriate object from context.
  - *The student* is working
  - $\exists y[\forall x[\text{student}'(x) \leftrightarrow x=y) \wedge \text{work}'(y)]$ (??)
- Utterances typically contain several noun phrases referring to different objects:
  - ***The student*** is reading ***the book*** in ***the library***
- Noun phrases may refer to different objects of the same type, in one utterance situation:
  - *the book*
  - *the blue book*

More context-dependent expressions

Semantic context dependence is a pervasive property of natural language:

(1) ***Every student*** *must be familiar with the basic properties of first-order logic*

(2) *It is hot and sunny* ***everywhere***

(3) *John* ***always*** *is late*

(4) *Bill has bought an* ***expensive*** *car*

(5) ***Another one****, please!*

A Problem with Indefinite NPs

a ⇒ $\lambda P \lambda Q \exists x[P(x) \wedge Q(x)]$

a student ⇒ $\lambda Q \exists x[\text{student}'(x) \wedge Q(x)]$

a student is working ⇒ $\exists x[\text{student}'(x) \wedge \text{work}'(x)]$

Indefinite Noun Phrases

- *A student is working*
  - $\Rightarrow \exists x[\text{student'}(x) \wedge \text{work'}(x)]$
  - *she* $\Rightarrow \lambda P.P(x)$
  - *She is successful* $\Rightarrow \text{successful'}(x)$
  - *A student is working. She is successful.*
    $\Rightarrow \exists x[\text{student'}(x) \wedge \text{work'}(x)] \wedge \text{successful'}(x)$
- Indefinite noun phrases establish the context for later reference, they introduce new reference objects:
  - ***A student*** *is working.* ***She*** *is successful.*
- Type-theoretic semantics cannot model this effect.

Discourse Semantics

- Natural-language meaning and context interact in two ways:
  - Context determines the utterance meaning.
  - The meaning of the utterance changes the context.
- The **„context change potential“** is part of the meaning of natural-language expressions.
- Division of labor between definite and indefinite NPs:
  - **Indefinite** NPs introduce **new** reference objects
  - **Definite** NPs refer to “old” or “**familiar**” reference objects

Discourse referents

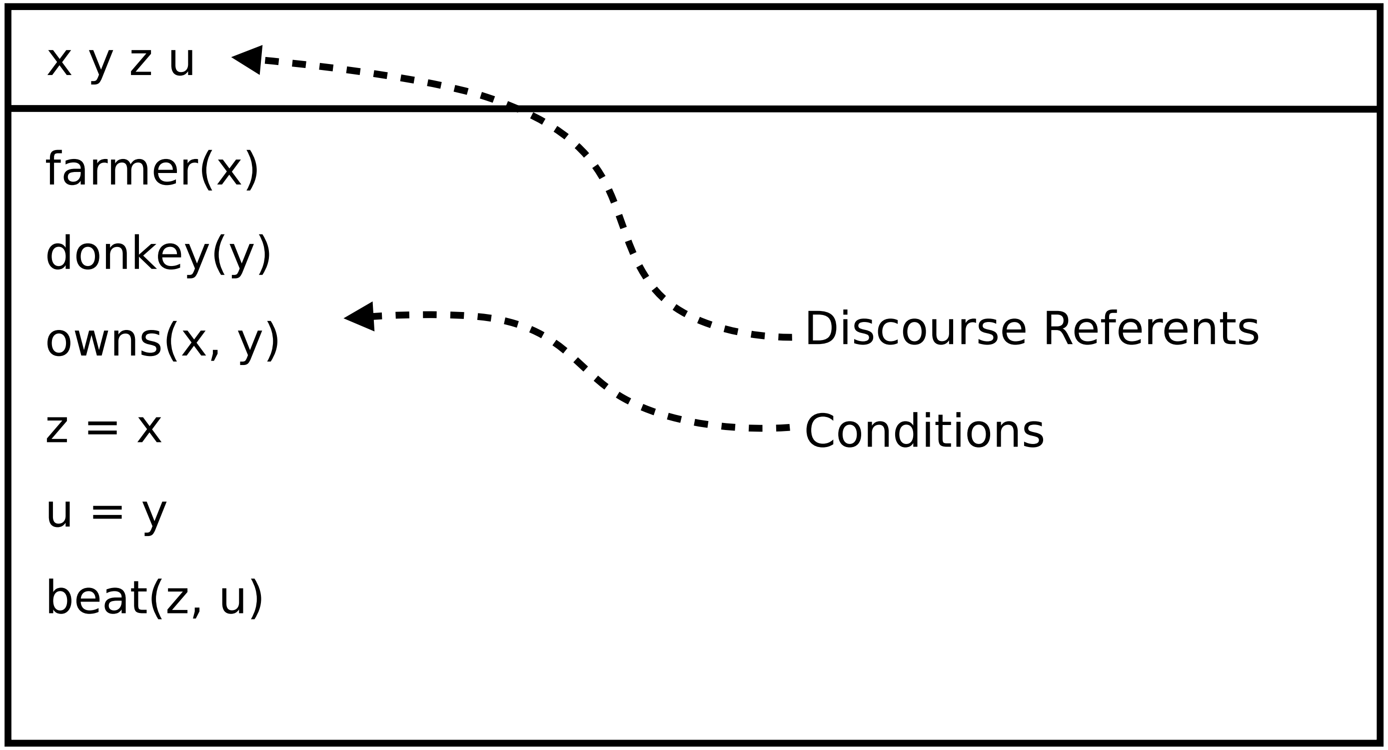
Reference objects established in discourse need not be specific entities:

If you have a pencil or a ball pen, could you please give it to me?

Someone – whoever that may be – will eventually find out. That person will tell others, and everyone will be terribly upset.

Discourse Representation Structures

- A farmer owns a donkey. He beats it.



DRS (Preliminary Version)

- **A discourse representation structure** (DRS) K is a pair $\langle U_K, C_K \rangle$, where
  - U_K is a set of discourse referents
  - C_K is a set of conditions
- **Conditions:**
  - $R(u_1, \ldots, u_n)$ R an n-place relation, $u_i \in U_K$
  - $u = v$ $u, v \in U_K$
  - $u = a$ $u \in U_K$, a is proper name

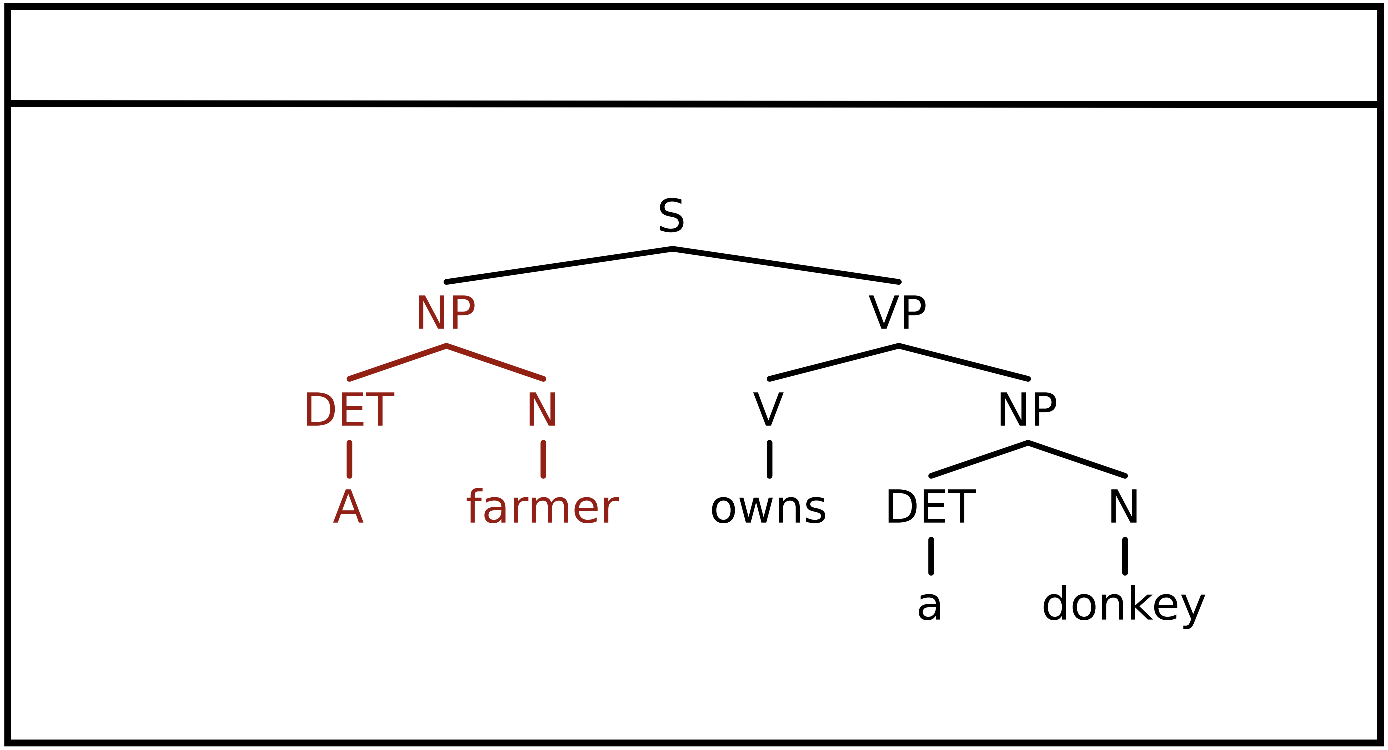
Discourse Representation Theory

Text	$\Sigma = \langle$	S_1,	S_2,	…,	S_n $\rangle$
		↓	↓		↓
Syntactic Analysis		$P(S_1)$	$P(S_2)$	…	$P(S_n)$
		↓	↓		↓
DRS Construction		**K_1** →	**K_2** →	**…** →	**K_n**
					↓

Interpretation by model embedding: Truth-conditions of Σ

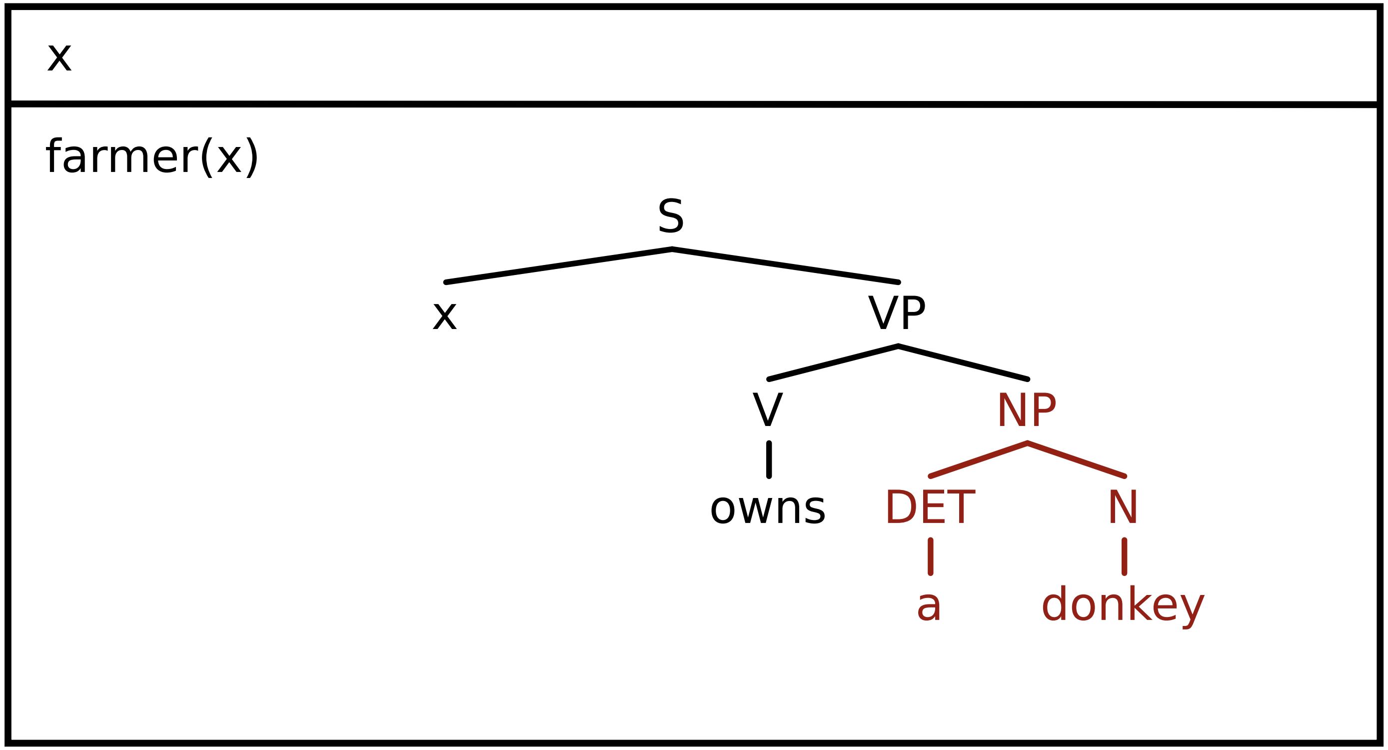
An Example

- A farmer owns a donkey. He beats it.



An Example

- A farmer owns a donkey. He beats it.



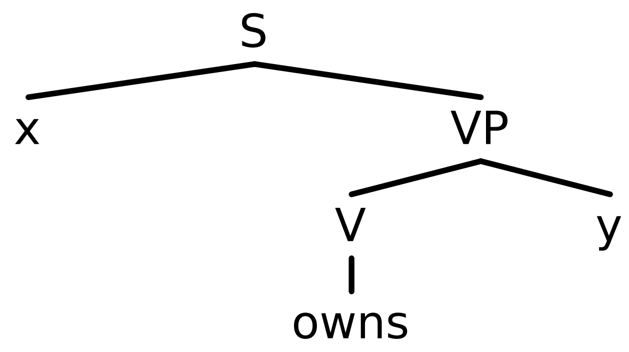
An Example

- A farmer owns a donkey. He beats it.

x y

farmer(x)

donkey(y)



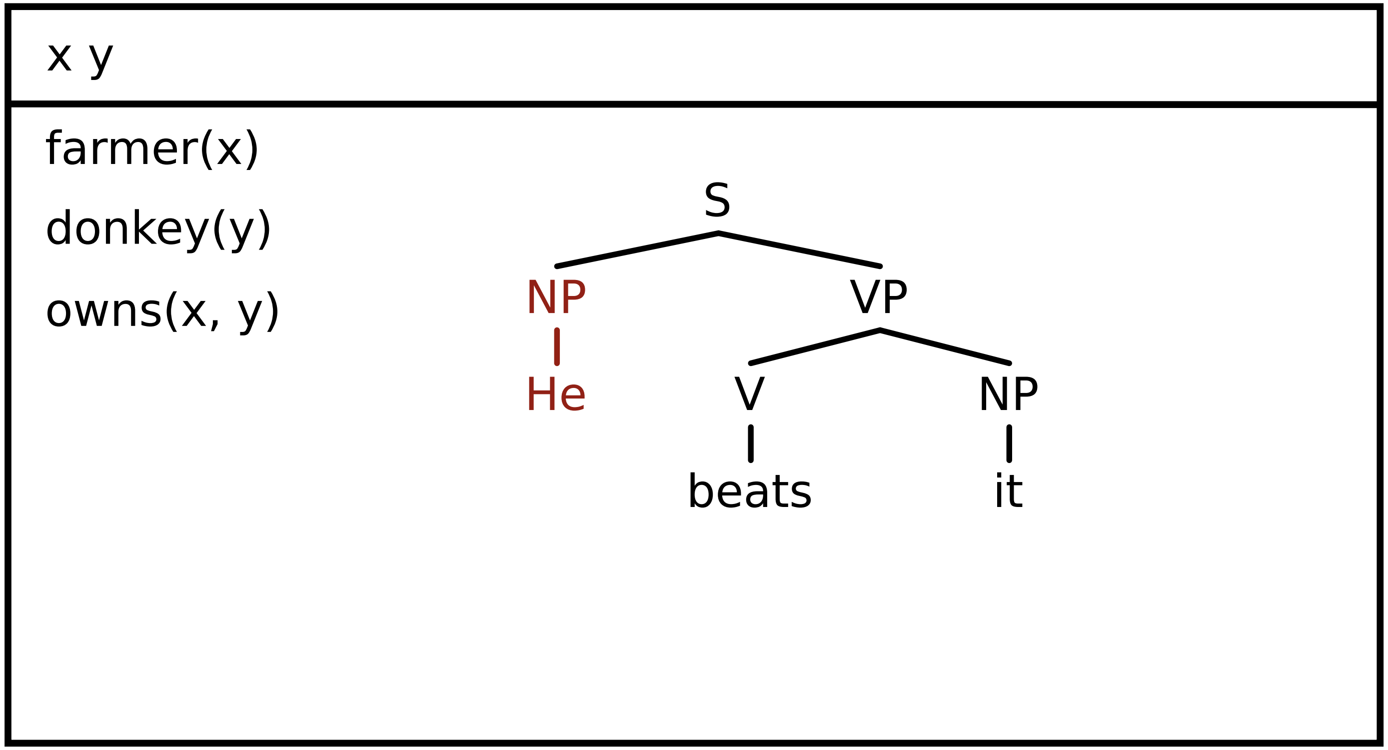
An Example

- A farmer owns a donkey. He beats it.

x y
farmer(x) donkey(y) owns(x, y)

An Example

- A farmer owns a donkey. He beats it.



An Example

- A farmer owns a donkey. He beats it.

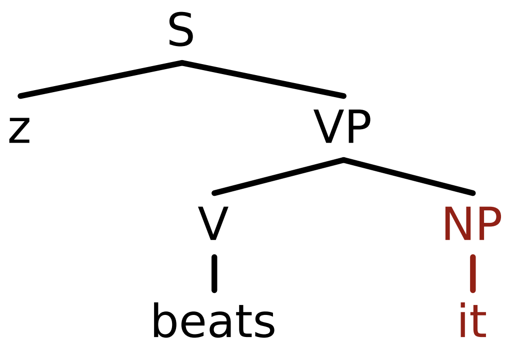
x y z

farmer(x)

donkey(y)

owns(x, y)

z = x



An Example

- A farmer owns a donkey. He beats it.

x y z u

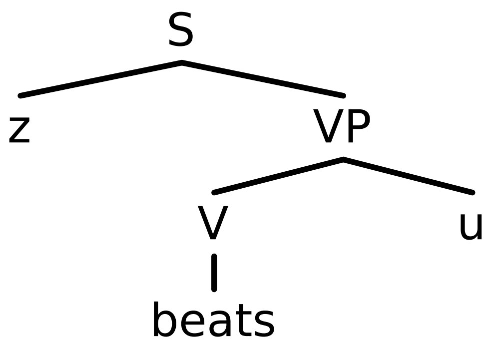
farmer(x)

donkey(y)

owns(x, y)

z = x

u = y



An Example

- A farmer owns a donkey. He beats it.

x y z u
farmer(x) donkey(y) owns(x, y) z = x u = y beat(z, u)

DRS (Basic Version)

- **A discourse representation structure** (DRS) K is a pair $\langle U_K, C_K \rangle$, where
  - U_K is a set of discourse referents
  - C_K is a set of (reduced of reducible) conditions
- **Reduced conditions:**
  - $R(u_1, \ldots, u_n)$ R an n-place relation, $u_i \in U_K$
  - $u = v$ $u, v \in U_K$
  - $u = a$ $u \in U_K$, a is proper name
- **Reducible conditions:**
  - Conditions of form α or $\alpha(x_1, \ldots, x_n)$, where α is a context-free parse tree.

DRS (Basic Version)

- A discourse referent (DR) **u is free in DRS K** = $\langle U_K, C_K \rangle$
  - if u is free in one of K's conditions,
  - and $u \notin U_K$.
- A DRS **K is closed** iff no DR occurs free in K.
- A reducible (fully reduced) DRS is a DRS which contains (does not contain) reducible conditions.

DRS Construction Algorithm

- Input:
  - a text $\Sigma = \langle S_1, \ldots, S_n \rangle$
  - a DRS K_0 (= $\langle \emptyset, \emptyset \rangle$, by default)
- Repeat for i = 1, …, n:
  - Add parse tree $P(S_i)$ to the conditions of K_{i-1}.
  - Apply DRS construction rules to reducible conditions of K_i, until no reduction steps are possible any more.
  - The resulting DRS is K_n , the discourse representation of text $\langle S_1, \ldots, S_n \rangle$.

Construction Rule for Indefinite NPs

- **Triggering Configuration:**
  - α is reducible condition in DRS K, containing $[_S\ [_{NP}\ \beta]\ [_{VP}\ \gamma]]$ or $[_{VP}\ [_V\ \gamma]\ [_{NP}\ \beta]]$ as a substructure.
  - β is εδ, ε indefinite article
- **Action:**
  - Add a new DR x to U_K.
  - Replace β in α by x.
  - Add δ(x) to C_K.

Construction Rule for Pronouns

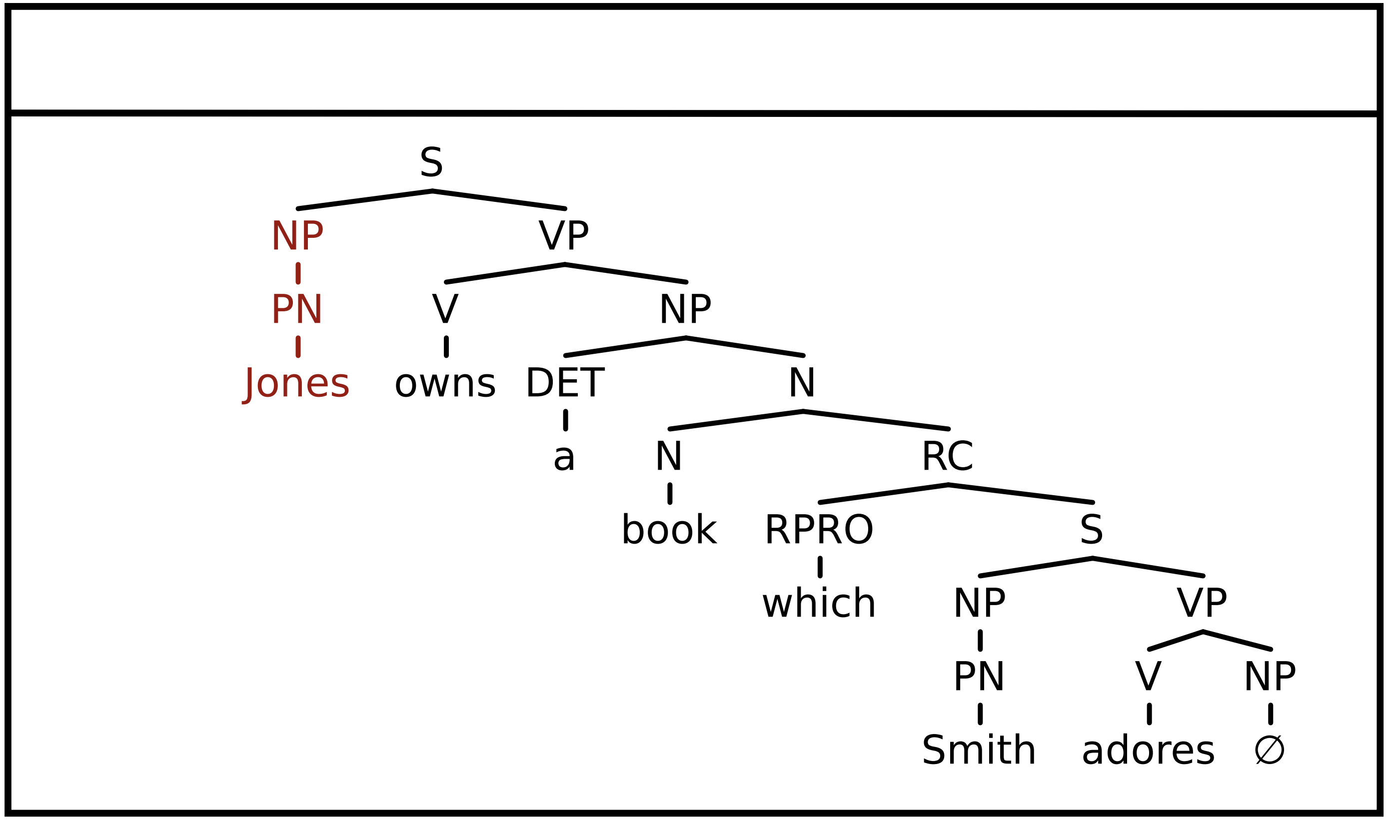
- **Triggering Configuration:**
  - α is reducible condition in DRS K, containing $[_S\ [_{NP}\ \beta]\ [_{VP}\ \gamma]]$ or $[_{VP}\ [_V\ \gamma]\ [_{NP}\ \beta]]$ as substructure.
  - β is a personal pronoun.
- **Action:**
  - Add a new DR x to U_K.
  - Replace β in α by x.
  - Select an appropriate DR $y \in U_K$, and add $x = y$ to C_K.

Construction Rule for Proper Names

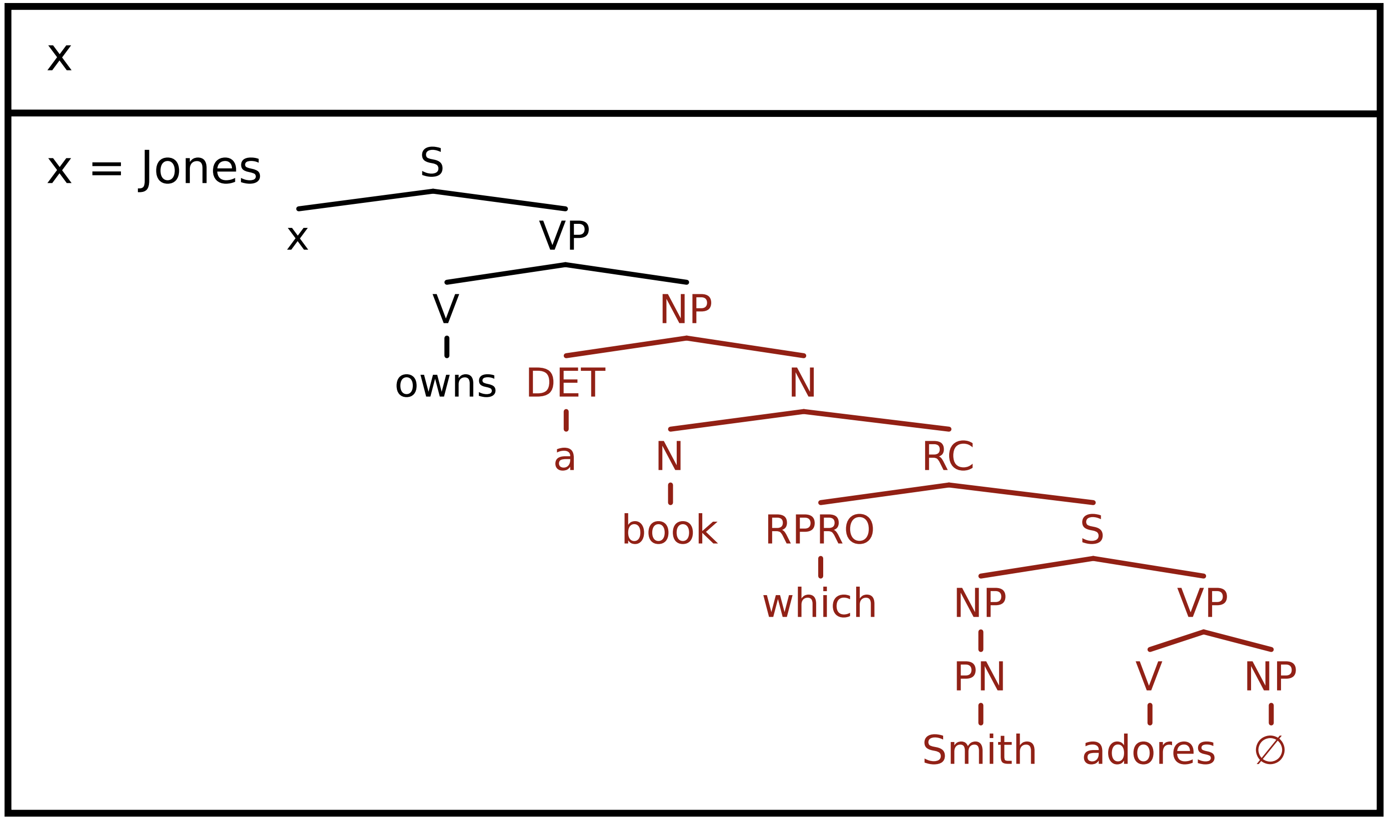
- **Triggering Configuration:**
  - α is reducible condition in DRS K, containing
    $[_{S} [_{NP}\ \beta] [_{VP}\ \gamma]]$ or $[_{VP} [_{V}\ \gamma] [_{NP}\ \beta]]$ as substructure.
  - β is a proper name.
- **Action:**
  - Add a new DR x to U_K.
  - Replace β in α by x.
  - Add x = β to C_K.

Relative Clauses

- Jones owns a book which Smith adores.



Relative Clauses

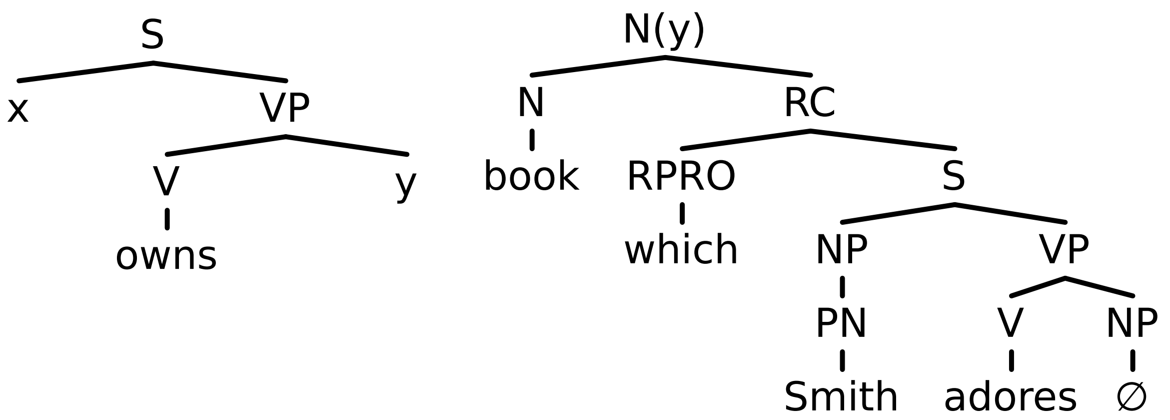


Relative Clauses

- Jones owns a book which Smith adores.

x y

x = Jones



Relative Clauses

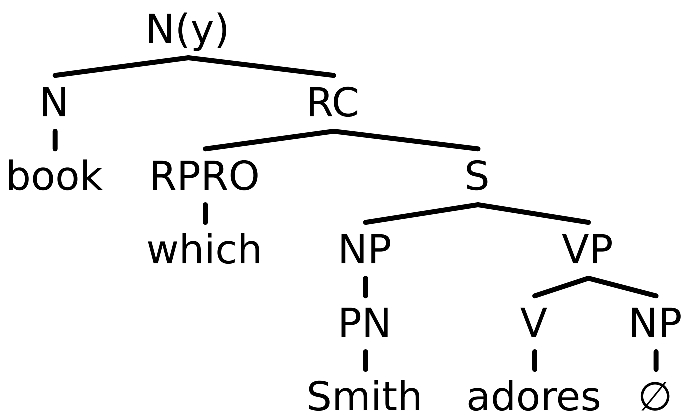
- Jones owns a book which Smith adores.

x y

---

x = Jones

owns(x, y)



Construction Rule for Relative

- **Triggering configuration:**
  - α(x) is reducible condition in DRS K, containing $[_{N'} [_{N'} \beta] [_{RC} \gamma]]$ as a substructure
  - γ is relative clause of the form δε, where δ is a relative pronoun and ε a sentence with an NP gap t, δ and t are co-indexed.
- **Actions:**
  - Remove α(x) from C_K.
  - Add β(x) to C_K .
  - Replace the NP gap in ε by x, and add the resulting structure to C_K.

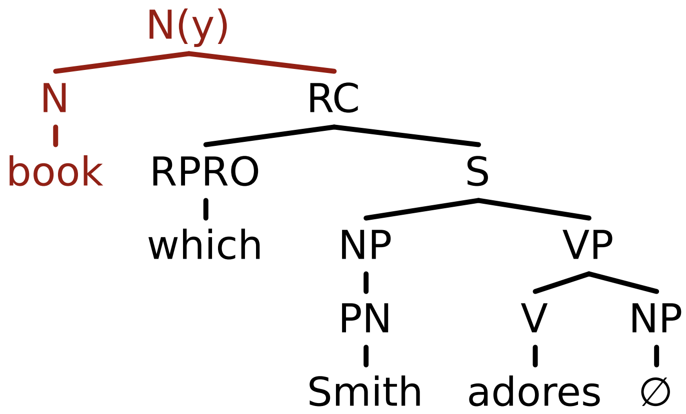
Relative Clauses

- Jones owns a book which Smith adores.

x y

x = Jones

owns(x, y)



Relative Clauses

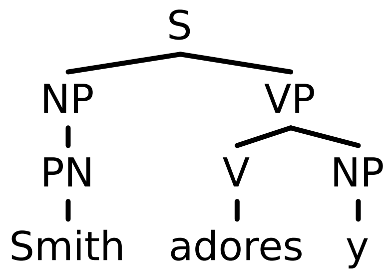
- Jones owns a book which Smith adores.

x y

x = Jones

owns(x, y)

book(y)



Relative Clauses

- Jones owns a book which Smith adores.

x y z
x = Jones owns(x, y) book(y) z = Smith adores(z, y)

A constraint on the DRS construction

A problem: The basic DRS construction algorithm can derive DRSes for both of the following sentences, with the indicated anaphoric binding:

(1) *[A professor]$_i$ recommends a book that she$_i$ likes*

(2) **She$_i$ recommends a book that [a professor]$_i$ likes*

The Highest Triggering Configuration

- If two triggering configurations of one or two different DRS construction rules occur in a reducible condition, then first apply the construction rule to the highest triggering configuration.
- The highest triggering configuration is the one whose top node dominates the top nodes of all other triggering configurations.

Discourse Representation Theory

Text	$\Sigma = \langle\ S_1,$	$S_2,$	$\ldots,$	$S_n\ \rangle$
	↓	↓		↓
Syntactic Analysis	$P(S_1)$	$P(S_2)$	$\ldots$	$P(S_n)$
	↓	↓		↓
DRS Construction	**K_1** →	**K_2** →	**…** →	**K_n**
				↓

Interpretation by model embedding: Truth-conditions of Σ

Denotational Interpretation

- Let
  - $K = \langle U_K, C_K \rangle$ a DRS
  - $M = \langle U_M, V_M \rangle$ a FOL model structure appropriate for K (i.e., M provides interpretations for all relation symbols occurring in K).
- An embedding of K into M is a function f from U_K to U_M.

Verifying embedding

- **An embedding** f of K in M **verifies K in M** iff f verifies every condition $\alpha \in C_K$
  - Notation: $f \models_M K$
- **f verifies condition α in M** ($f \models_M \alpha$):
  - $f \models_M R(x_1, \ldots, x_n)$ iff $\langle f(x_1), \ldots, f(x_n) \rangle \in V_M(R)$
  - $f \models_M x = a$ iff $f(x) = V_M(a)$
  - $f \models_M x = y$ iff $f(x) = f(y)$

Truth

- Let K be a closed DRS and M be an appropriate model structure for K.
- K is true in M iff there is a verifying embedding f of K in M such that Dom(f) = U_K