Discourse Structure in Dialogue

Lecture 2: Discourse and Coherence

Julian J. Schlöder

DRT

The Syntax–Semantics Interface (simple)

- "x is P" $\rightsquigarrow P(x)$
- "x is y" $\rightsquigarrow x = y$
- "an x is P" $\rightsquigarrow \exists x.(P(x))$
- "all x are P" $\rightsquigarrow \forall x.(P(x))$
- "all P are Q" $\rightsquigarrow \forall x.(P(x) \rightarrow Q(x))$
- "If p, then q" $\rightsquigarrow p \rightarrow q$

The Syntax–Semantics Interface (simple)

- "x is P" $\rightsquigarrow P(x)$
- "x is y" $\rightsquigarrow x = y$
- "an x is P" $\rightsquigarrow \exists x.(P(x))$
- "all x are P" $\rightsquigarrow \forall x.(P(x))$
- "all P are Q" $\rightsquigarrow \forall x.(P(x) \rightarrow Q(x))$
- "If p, then q" $\rightsquigarrow p \rightarrow q$
- "If x is y and x is P, then y is P" $\rightsquigarrow (x = y \wedge P(x)) \rightarrow P(y)$.
- "If all x are P and a y is Q, then an x is P and Q"
  $\rightsquigarrow (\forall x.(P(x)) \wedge \exists y.(Q(y))) \rightarrow \exists x.(P(x) \wedge Q(x))$.

Enter Anaphora

Enter Anaphora

– It *should* be simple to integrate “he”, “she”, “it” *etc.*

Enter Anaphora

- It *should* be simple to integrate “he”, “she”, “it” *etc.*
- Donkey Sentences (Geach, 1962)

(1) Every farmer who owns a donkey beats it.

Enter Anaphora

- It *should* be simple to integrate “he”, “she”, “it” *etc.*
- Donkey Sentences (Geach, 1962)

(1) Every farmer who owns a donkey beats it.

✗ $\forall x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x, y)) \rightarrow \texttt{beat}(x, {}^{\#}y)$

Enter Anaphora

- It *should* be simple to integrate “he”, “she”, “it” *etc.*
- Donkey Sentences (Geach, 1962)

(1) Every farmer who owns a donkey beats it.

✗ $\forall x.(\mathtt{farmer}(x) \wedge \exists y.(\mathtt{donkey}(y) \wedge \mathtt{own}(x, y)) \rightarrow \mathtt{beat}(x, {}^{\#}y)$

✗ $\forall x \exists y.((\mathtt{farmer}(x) \wedge \mathtt{donkey}(y) \wedge \mathtt{own}(x, y)) \rightarrow \mathtt{beat}(x, y))$

Enter Anaphora

- It *should* be simple to integrate "he", "she", "it" *etc.*
- Donkey Sentences (Geach, 1962)

(1) Every farmer who owns a donkey beats it.

✗ $\forall x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x, {}^{\#}y)$

✗ $\forall x \exists y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

✓ $\forall x \forall y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

Enter Anaphora

- It *should* be simple to integrate "he", "she", "it" *etc.*
- Donkey Sentences (Geach, 1962)

(1) Every farmer who owns a donkey beats it.

✗ $\forall x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x, {}^{\#}y)$

✗ $\forall x \exists y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

✓ $\forall x \forall y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

- BUT:

(2) Every farmer who owns a donkey (also) owns a pig.

✓ $\forall x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x,y))$
$\rightarrow \exists y.(\texttt{pig}(y) \wedge \texttt{own}(x,y))$

Donkey Sentences (today)

(3) If a farmer owns a donkey, he beats it.

Donkey Sentences (today)

(3) If a farmer owns a donkey, he beats it.

✗ $(\exists x.(\mathtt{farmer}(x) \wedge \exists y.(\mathtt{donkey}(y) \wedge \mathtt{own}(x, y)))) \rightarrow \mathtt{beat}({}^{\#}x, {}^{\#}y)$

Donkey Sentences (today)

(3) If a farmer owns a donkey, he beats it.

✗ $(\exists x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x, y)))) \rightarrow \texttt{beat}(^{\#}x, ^{\#}y)$

✗ $\exists x \exists y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x, y)) \rightarrow \texttt{beat}(x, y))$

Donkey Sentences (today)

(3) If a farmer owns a donkey, he beats it.

✗ $(\exists x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x, y)))) \rightarrow \texttt{beat}(^{\#}x, ^{\#}y)$

✗ $\exists x \exists y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x, y)) \rightarrow \texttt{beat}(x, y))$

✓ $\forall x \forall y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x, y)) \rightarrow \texttt{beat}(x, y))$

Donkey Sentences (today)

(3) If a farmer owns a donkey, he beats it.

✗ $(\exists x.(\texttt{farmer}(x) \wedge \exists y.(\texttt{donkey}(y) \wedge \texttt{own}(x,y)))) \rightarrow \texttt{beat}(^{\#}x, ^{\#}y)$

✗ $\exists x \exists y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

✓ $\forall x \forall y.((\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)) \rightarrow \texttt{beat}(x,y))$

– There's really no end to this.

(4) If a donkey is not beaten, it's happy.

A farmer beats his donkeys.

If someone loves something, he won't beat it.

Storyboard Again

- The idea:
- “If a farmer owns a donkey, he beats it” is a bit like Whenever a farmer owns a donkey, he beats it.
- Better yet, write *Whenever the situation [the current scene in the story] is such that it contains a farmer, a donkey and the farmer owns the donkey, then the situation [scene] is such that the farmer beats the donkey.*
- Or *for every situation...*

Discourse Representation Theory

- The sentence “He beats it” does not really have truth-conditions.
- Rather, it is embedded in a discourse and only has definite truth-conditions when interpreted in that discourse.

Discourse Representation Theory

- The sentence “He beats it” does not really have truth-conditions.
- Rather, it is embedded in a discourse and only has definite truth-conditions when interpreted in that discourse.
- Information states K are structures *representing* the contents of a discourse under interpretation.
- And these *representations* can be *evaluated* (truth-conditionally, if you like).
- If we cannot construct a representation, then the discourse is incoherent.

DRT Architecture

– $\boxed{\text{Natural Language Sentences}} \rightsquigarrow \boxed{\text{DR Structures}} \mapsto \boxed{\text{First Order Models}}$,

where:

$\rightsquigarrow :=$ the discourse representation *construction algorithm*,
$\mapsto :=$ a truth-conditional model-theoretic *evaluation*.

Discourse Representation Structures

- Discourse representation structures (DRSs) function as partial descriptions of particular situations/scenes.

- They consist of a universe of things and a set of conditions attributed to them.
  $\rightarrow$ $K := \langle U, \mathrm{CON} \rangle$.

$$\begin{array}{|c|}\hline x_1, x_2, \ldots, x_n \\ \hline \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_n \\ \hline \end{array}$$

$\leftarrow$ the things we talk about

$\leftarrow$ what we say about these things

The Language of DRSs

$$\text{for } x \in \mathrm{VAR}, N \in \mathrm{NAME}, P^k \in \mathrm{REL},$$

$$\mathrm{Con} := N(x) \mid P^k(x_1, \ldots, x_k) \mid x_i = x_j \mid \neg K \mid K \vee K \mid K \Rightarrow K,$$

$$K := \begin{array}{|c|} \hline x_1, \ldots, x_n \\ \hline \mathrm{Con}_1 \\ \vdots \\ \mathrm{Con}_m \\ \hline \end{array}$$

$$(= \langle \{x_1, \ldots, x_n\}, \{\mathrm{Con}_1, \ldots, \mathrm{Con}_m\} \rangle)$$

A Simple DRS

- "A farmer beats a donkey."
- $\langle \{f, d\}, \{\texttt{farmer}(f), \texttt{donkey}(d), \texttt{beat}(f, d)\}\rangle$.
- $$\begin{array}{|c|}\hline \text{f,d} \\ \hline \texttt{farmer}(f) \\ \texttt{donkey}(d) \\ \texttt{beat}(f, d) \\ \hline \end{array}$$

A Less Simple DRS

- "If a farmer owns a donkey, he beats it."
- $$\begin{array}{|c|}\hline \\ \hline \begin{array}{|c|}\hline f, d \\ \hline \texttt{farmer}(f) \\ \texttt{donkey}(d) \\ \texttt{owns}(f, d) \\ \hline \end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline \texttt{beat}(f, d) \\ \hline \end{array} \\ \hline \end{array}$$

every vs *some*

"A man walks."

$$\begin{array}{|c|}\hline x \\ \hline \texttt{man}(x) \\ \texttt{walks}(x) \\ \hline \end{array}$$

"Every man walks."

$$\begin{array}{|c|}\hline \\ \hline \begin{array}{|c|}\hline x \\ \hline \texttt{man}(x) \\ \hline \end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline \texttt{walk}(x) \\ \hline \end{array} \\ \hline \end{array}$$

Accessibility (not as bad as it looks)

A DRS K is **immediately accessible** to a DRS K' iff:

1. K contains the condition $\neg K'$; or
2. K contains a condition of the form $K' \bigvee K''$ or $K'' \bigvee K'$.
3. K contains a condition of the form $K' \Rightarrow K''$.
4. There is a DRS K'' that contains the condition $K \Rightarrow K'$.

Accessibility (not as bad as it looks)

A DRS K is **immediately accessible** to a DRS K' iff:

1. K contains the condition $\neg K'$; or
2. K contains a condition of the form $K' \bigvee K''$ or $K'' \bigvee K'$.
3. K contains a condition of the form $K' \Rightarrow K''$.
4. There is a DRS K'' that contains the condition $K \Rightarrow K'$.

K is **accessible** to K' if K' is connected to K via immediate accessibility ("up or left in conditionals").

That is, if there is a chain $K' = K_1, K_2, ..., K_{n-1}, K_n = K$ where for all i, K_i is immediately accessible to K_{i-1}.

- Now, a pronoun in K' can access referents in the universes of all DRSs K that are accessible to K'.

Accessibility (Negation)

(5) It is not the case that a man is running. #He takes his time.

$$
\boxed{\begin{array}{c}
\\ \hline
\neg\ \boxed{\begin{array}{c} x \\ \hline \mathtt{man}(x) \\ \mathtt{runs}(x) \end{array}} \\
\mathtt{take\_time}({}^{?}x)
\end{array}}
$$

Accessibility (Negation)

(5) It is not the case that a man is running. #He takes his time.

$$\begin{array}{|c|}\hline \\ \hline \neg \begin{array}{|c|}\hline x \\ \hline \texttt{man}(x) \\ \texttt{runs}(x) \\ \hline\end{array} \\ \texttt{take\_time}(^{?}x) \\ \hline\end{array}$$

– This is actually fine, but for different reasons (later!):

(6) It is not the case that John is running. He takes his time.

Accessibility (Negation)

(7) A man is not running. He takes his time.

$$
\begin{array}{|c|}
\hline
x \\
\hline
\texttt{man}(x) \\
\neg \begin{array}{|c|}
\hline
\\
\hline
\texttt{runs}(x) \\
\hline
\end{array} \\
\texttt{take\_time}(x) \\
\hline
\end{array}
$$

Accessibility (Negation)

(8) Not every man is running. #He takes his time.

$$\begin{array}{|c|}\hline \\ \hline \neg \begin{array}{|c|}\hline \\ \hline \begin{array}{|c|}\hline x \\ \hline \texttt{man}(x) \\ \hline\end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline \texttt{runs}(x) \\ \hline\end{array} \\ \hline \end{array} \\ \texttt{take\_time}(^{?}x) \\ \hline \end{array}$$

Accessibility (Disjunction)

– Can't go left or right in disjunction (controversial).

(9) Either a man is having tea or ${}^{?}$he is having coffee.

$$
\begin{array}{|c|}
\hline
\\
\hline
\begin{array}{|c|}\hline x, y \\ \hline \texttt{man}(x) \\ \texttt{tea}(z) \\ \texttt{have}(x, z) \\ \hline\end{array} \bigvee \begin{array}{|c|}\hline z \\ \hline \texttt{coffee}(z) \\ \texttt{have}({}^{?}x, z) \\ \hline\end{array} \\
\hline
\end{array}
$$

Accessibility (Disjunction)

– Can't go left or right in disjunction (controversial).

(9) Either a man is having tea or $^{?}$he is having coffee.

$$\boxed{\begin{array}{c} \\ \hline \boxed{\begin{array}{c} x, y \\ \hline \texttt{man}(x) \\ \texttt{tea}(z) \\ \texttt{have}(x, z) \end{array}} \vee \boxed{\begin{array}{c} z \\ \hline \texttt{coffee}(z) \\ \texttt{have}(^{?}x, z) \end{array}} \end{array}}$$

– This is actually fine, but for different reasons:

(10) Either John is having tea or he is having coffee.

Interpretation of DRSs: Referent Extension

- The idea is this: DRSs tell us something about who’s who.
- Every DRS can introduce new referents and impose conditions on (new or previous) referents.

Interpretation of DRSs: Referent Extension

- The idea is this: DRSs tell us something about who's who.
- Every DRS can introduce new referents and impose conditions on (new or previous) referents.

Referent System Extension

Let $M = (D_M, I_M)$ be a FOL model. Let $f, g : \text{VAR} \rightarrow D_M$ be partial functions (referent systems). Let $K = \langle U, Conds \rangle$ be a DRS.

- $f[K]_M g$ iff $\text{dom}(g) = \text{dom}(f) \cup U$; and
  $\forall x \in \text{dom}(f) : f(x) = g(x)$.

Interpretation of DRSs: Truth 1

– The conditions on the referents impose tests.

DRS Semantics

By simultaneous recursion:

1. $f[\![\langle U, Cons\rangle]\!]_M g$ iff $f[\langle U, Cons\rangle]_M g$ and $M, g \models_{DRT} C$ for all $C \in Cons$.

Interpretation of DRSs: Truth 1

– The conditions on the referents impose tests.

DRS Semantics

By simultaneous recursion:

1. $f[\![\langle U, Cons\rangle]\!]_M g$ iff $f[\langle U, Cons\rangle]_M g$ and $M, g \models_{DRT} C$ for all $C \in Cons$.
   a. $M, f \models_{DRT} x = y$ iff $f(x) = f(y)$.
   b. $M, f \models_{DRT} N(x)$ iff $M, f \models N(f(x))$.
   c. $M, f \models_{DRT} R(x_1, \cdots, x_n)$ iff $M, f \models R(f(x_1), \cdots, f(x_n))$.

Interpretation of DRSs: Truth 1

– The conditions on the referents impose tests.

DRS Semantics

By simultaneous recursion:

1. $f[\![\langle U, Cons\rangle]\!]_M g$ iff $f[\langle U, Cons\rangle]_M g$ and $M, g \models_{DRT} C$ for all $C \in Cons$.
   a. $M, f \models_{DRT} x = y$ iff $f(x) = f(y)$.
   b. $M, f \models_{DRT} N(x)$ iff $M, f \models N(f(x))$.
   c. $M, f \models_{DRT} R(x_1, \cdots, x_n)$ iff $M, f \models R(f(x_1), \cdots, f(x_n))$.
2. $M, f \models_{DRT} \neg K$ iff there is no g with $f[\![K]\!]_M g$.
3. $M, f \models_{DRT} K_1 \vee K_2$ iff there is a g with $f[\![K_1]\!]_m g$ or $f[\![K_2]\!]_M g$.
4. $M, f \models_{DRT} K_1 \Rightarrow K_2$ iff for every g with $f[\![K_1]\!]_M g$ there is a h with $g[\![K_2]\!]_M h$.

Donkey Sentences

- “If a farmer owns a donkey, he beats it.”
- $$\begin{array}{|c|}\hline \\ \hline \begin{array}{|c|}\hline f, d \\ \hline \texttt{farmer}(f) \\ \texttt{donkey}(d) \\ \texttt{owns}(f, d) \\ \hline \end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline \texttt{beat}(f, d) \\ \hline \end{array} \\ \hline \end{array}$$
- $\approx$ Whenever we have a farmer and we have a donkey and the farmer owns the donkey, then the farmer beats the donkey.

Interpretation of DRSs: Truth 2

– DRT embeds into classical (Tarskian) First Order Semantics.

DRT embeds into FOL

Define recursively:

–
$$
\begin{aligned}
(P^k(x_1,\ldots,x_k))^{\mathrm{FOL}} &= P^k x_1,\ldots,x_k;\\
(x_i = x_j)^{\mathrm{FOL}} &= (x_i = x_j);\\
(\neg K)^{\mathrm{FOL}} &= \neg K^{\mathrm{FOL}};\\
(K_1 \vee K_2)^{\mathrm{FOL}} &= (K_1^{\mathrm{FOL}} \vee K_2^{\mathrm{FOL}});
\end{aligned}
$$

– If $K_1 = \langle\{x_1,\ldots,x_n\},\{\mathrm{Con}_1,\ldots,\mathrm{Con}_m\}\rangle$, then

$$
\begin{aligned}
K_1^{\mathrm{FOL}} &= \exists x_1 \ldots \exists x_n(\mathrm{Con}_1^{\mathrm{FOL}} \wedge \ \ldots \ \wedge \mathrm{Con}_m^{\mathrm{FOL}});\\
(K_1 \Rightarrow K_2)^{\mathrm{FOL}} &= \forall x_1 \ldots \forall x_n((\mathrm{Con}_1^{\mathrm{FOL}} \wedge \ \ldots \ \wedge \mathrm{Con}_m^{\mathrm{FOL}}) \rightarrow K_2^{\mathrm{FOL}}).
\end{aligned}
$$

Interpretation of DRSs: Truth 2

– DRT embeds into classical (Tarskian) First Order Semantics.

DRT embeds into FOL

Define recursively:

– $$\begin{aligned}
(P^k(x_1,\ldots,x_k))^{\mathrm{FOL}} &= P^k x_1,\ldots,x_k;\\
(x_i = x_j)^{\mathrm{FOL}} &= (x_i = x_j);\\
(\neg K)^{\mathrm{FOL}} &= \neg K^{\mathrm{FOL}};\\
(K_1 \vee K_2)^{\mathrm{FOL}} &= (K_1^{\mathrm{FOL}} \vee K_2^{\mathrm{FOL}});
\end{aligned}$$

– If $K_1 = \langle\{x_1,\ldots,x_n\},\{\mathrm{Con}_1,\ldots,\mathrm{Con}_m\}\rangle$, then

$$\begin{aligned}
K_1^{\mathrm{FOL}} &= \exists x_1 \ldots \exists x_n(\mathrm{Con}_1^{\mathrm{FOL}} \wedge \ \ldots \ \wedge \mathrm{Con}_m^{\mathrm{FOL}});\\
(K_1 \Rightarrow K_2)^{\mathrm{FOL}} &= \forall x_1 \ldots \forall x_n((\mathrm{Con}_1^{\mathrm{FOL}} \wedge \ \ldots \ \wedge \mathrm{Con}_m^{\mathrm{FOL}}) \rightarrow K_2^{\mathrm{FOL}}).
\end{aligned}$$

– For all f: $M,f \models_{DRT} K$ iff $M,f \models_{FOL} K^{\mathrm{FOL}}$.

DRS Construction Algorithm (Kamp and Reyle 1993)

DRS-Construction Algorithm

Input: a discourse D = $S_1, \ldots, S_i, S_{i+1}, \ldots, S_n$
the empty DRS K_0

Keep repeating for $i = 1, \ldots, n$**:**

(i) add the syntactic analysis $[S_i]$ of (the next) sentence S_i to the conditions of K_{i-1}; call this DRS K_i*. Go to (ii).

(ii) Input: a set of reducible conditions of K_i*
Keep on applying construction principles to each reducible condition of K_i* until a DRS K_i is obtained that only contains irreducible conditions. Go to (i).

– (It's a *shift-reduce* algorithm, in case that means something to someone.)

DRT-style discourse

S-DRT will do away with this simple construction!

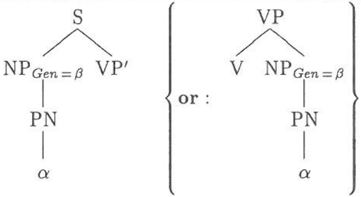
– $$\boxed{\begin{array}{c} x_1, \ldots, x_n \\ \hline \phi_1 \\ \vdots \\ \phi_m \end{array}} \bigoplus S, S', \ldots \rightsquigarrow \boxed{\begin{array}{c} x_1, \ldots, x_n \\ \hline \phi_1 \\ \vdots \\ \phi_m \\ [\mathrm{S}] \end{array}} \bigoplus S', \ldots$$

– If this stops before all S have been dealt with, the discourse is uninterpretable.

DRS Construction Algorithm: Names

CR.PN

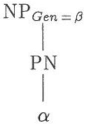
Triggering configuration $\gamma \subseteq \overline{\gamma} \in \mathbf{Con_K}$:



Introduce into the universe of the main DRS: new discourse referent $\mathbf{u}$

Introduce into the condition set of the main DRS: new conditions $\alpha(\mathbf{u})$, $Gen(\mathbf{u}) = \beta$

Substitute in $\overline{\gamma}$: $\mathbf{u}$ for



Example: Names

Julian smiled. He saw a cat.

$\rightsquigarrow \boxed{\begin{array}{c} \\ \hline \\ \end{array}} \oplus$ Julian smiled $\oplus$ He saw a cat.

$\rightsquigarrow \boxed{\begin{array}{c} \\ \hline \text{Julian smiled.} \end{array}} \oplus$ He saw a cat.

Example: Names

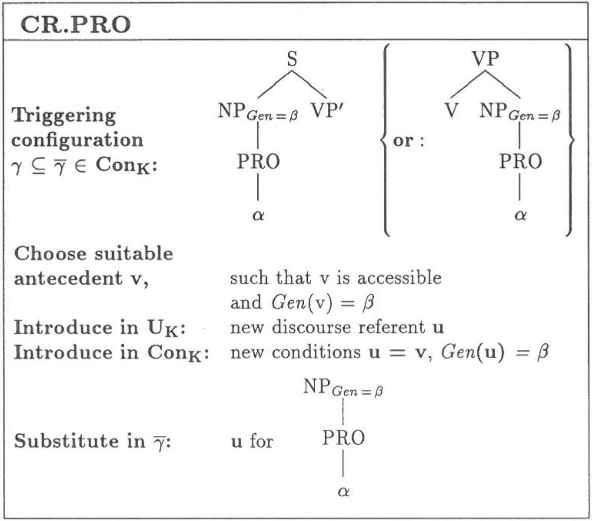
Julian smiled. He saw a cat.

$$\rightsquigarrow \begin{array}{|c|}\hline \\ \hline \\ \\ \hline \end{array} \oplus \text{Julian smiled} \oplus \text{He saw a cat.}$$

$$\rightsquigarrow \begin{array}{|c|}\hline \\ \hline \text{Julian smiled.} \\ \hline \end{array} \oplus \text{He saw a cat.}$$

$$\rightsquigarrow \begin{array}{|c|}\hline j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ j \text{ smiled.} \\ \hline \end{array} \oplus \text{He saw a cat.}$$

DRS Construction Algorithm: Pronouns



CR.PRO

Triggering configuration $\gamma \subseteq \overline{\gamma} \in \mathbf{Con_K}$**:**	$[_{S}\ [_{NP_{Gen=\beta}}\ [_{PRO}\ \alpha]]\ VP']$ **or :** $[_{VP}\ V\ [_{NP_{Gen=\beta}}\ [_{PRO}\ \alpha]]]$
Choose suitable antecedent v,	such that v is accessible and $Gen(\mathrm{v}) = \beta$
Introduce in $\mathbf{U_K}$**:**	new discourse referent **u**
Introduce in $\mathbf{Con_K}$**:**	new conditions $\mathbf{u} = \mathbf{v}$, $Gen(\mathbf{u}) = \beta$
Substitute in $\overline{\gamma}$**:**	**u** for $[_{NP_{Gen=\beta}}\ [_{PRO}\ \alpha]]$

Example: Pronouns

$$\boxed{\begin{array}{c} j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \end{array}} \oplus \text{He saw a cat.}$$

Example: Pronouns

$$\boxed{\begin{array}{c} j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \end{array}} \oplus \text{He saw a cat.}$$

$$\rightsquigarrow \boxed{\begin{array}{c} j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \\ \text{He saw a cat.} \end{array}}$$

Example: Pronouns

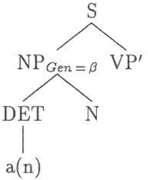
$$\boxed{\begin{array}{c} j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \end{array}} \oplus \text{He saw a cat.}$$

$$\rightsquigarrow \boxed{\begin{array}{c} j \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \\ \text{He saw a cat.} \end{array}} \quad \rightsquigarrow \quad \boxed{\begin{array}{c} j, u \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \\ \text{u} = \text{j} \\ \text{Gen}(u) = \text{m} \\ \text{u saw a cat.} \end{array}}$$

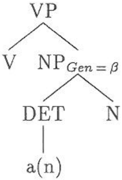
DRS Construction Algorithm: Indefinites

CR.ID

Triggering configuration $\gamma \subseteq \overline{\gamma} \in$ **Con$_{\mathbf{K}}$**:



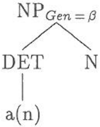
or :



Introduce in U$_{\mathbf{K}}$: new discourse referent **u**

Introduce in Con$_{\mathbf{K}}$: new condition **[N](u)**, **Gen(u)**=β

Substitute in $\overline{\gamma}$: **u** for



Example: Indefinites

j,u
Julian(j)
Gen(*j*) = m
`smile`(*j*)
u = j
Gen(*u*) = m
u saw a cat.

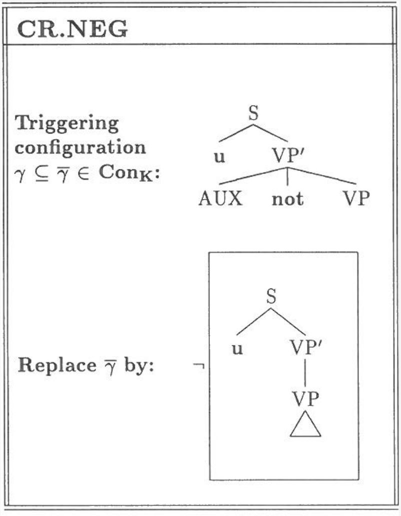
Example: Indefinites

$$\begin{array}{|c|}\hline \text{j,u} \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \\ \text{u} = \text{j} \\ \text{Gen}(u) = \text{m} \\ \text{u saw a cat.} \\ \hline \end{array} \quad \rightsquigarrow \quad \begin{array}{|c|}\hline \text{j,u,v} \\ \hline \text{Julian}(j) \\ \text{Gen}(j) = \text{m} \\ \texttt{smile}(j) \\ \text{u} = \text{j} \\ \text{Gen}(u) = \text{m} \\ \text{Gen}(v) = \text{n} \\ [\text{cat}](v) \\ \text{u saw } v\text{.} \\ \hline \end{array}$$

Example: Indefinites

$$
\begin{array}{|c|}
\hline
\text{j,u} \\
\hline
\text{Julian}(j) \\
\text{Gen}(j) = \text{m} \\
\texttt{smile}(j) \\
\text{u} = \text{j} \\
\text{Gen}(u) = \text{m} \\
\text{u saw a cat.} \\
\hline
\end{array}
\rightsquigarrow
\begin{array}{|c|}
\hline
\text{j,u,v} \\
\hline
\text{Julian}(j) \\
\text{Gen}(j) = \text{m} \\
\texttt{smile}(j) \\
\text{u} = \text{j} \\
\text{Gen}(u) = \text{m} \\
\text{Gen}(v) = \text{n} \\
\text{[cat]}(v) \\
\text{u saw } v. \\
\hline
\end{array}
\rightsquigarrow
\begin{array}{|c|}
\hline
\text{j,u,v} \\
\hline
\text{Julian}(j) \\
\text{Gen}(j) = \text{m} \\
\texttt{smile}(j) \\
\text{u} = \text{j} \\
\text{Gen}(u) = \text{m} \\
\text{Gen}(v) = \text{n} \\
\texttt{cat}(v) \\
\text{u saw } v. \\
\hline
\end{array}
$$

DRS Construction Algorithm: Negation



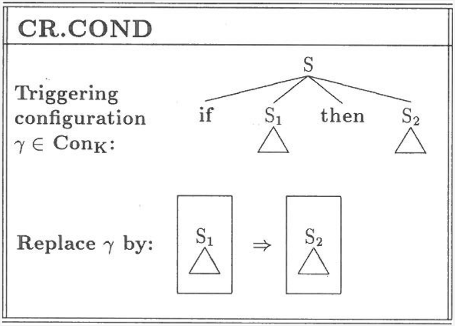
Example: Negation

(11) A man is not seeing a cat. He smiles, #it does not.

$$
\begin{array}{|c|}
\hline
\\
\hline
\text{A man is not seeing a cat} \\
\hline
\end{array}
\rightsquigarrow
\begin{array}{|c|}
\hline
x \\
\hline
\mathtt{man}(x) \\
\neg \begin{array}{|c|}
\hline
\\
\hline
x \text{ sees a cat} \\
\hline
\end{array} \\
\hline
\end{array}
$$

$$
\rightsquigarrow
\begin{array}{|c|}
\hline
x \\
\hline
\mathtt{man}(x) \\
\neg \begin{array}{|c|}
\hline
y \\
\hline
\mathtt{cat}(y) \\
\mathtt{see}(x, y) \\
\hline
\end{array} \\
\hline
\end{array}
$$

DRS Construction Algorithm: Conditionals

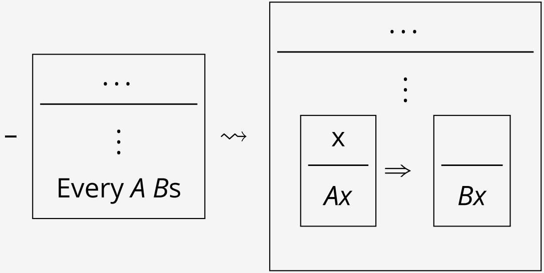


(12) If a farmer owns a donkey, he beats it.

$$
\begin{array}{|c|}
\hline
\\
\hline
\begin{array}{|c|}
\hline
\\
\hline
\text{A farmer owns a donkey} \\
\hline
\end{array}
\Rightarrow
\begin{array}{|c|}
\hline
\\
\hline
\text{He beats it} \\
\hline
\end{array} \\
\hline
\end{array}
\rightsquigarrow
$$

$$
\begin{array}{|c|}
\hline
\\
\hline
\begin{array}{|c|}
\hline
x, y \\
\hline
\mathtt{farmer}(x) \\
\mathtt{donkey}(y) \\
\mathtt{owns}(x, y) \\
\hline
\end{array}
\Rightarrow
\begin{array}{|c|}
\hline
x', y' \\
\hline
\mathtt{beat}(x', y') \\
x' = x \\
y' = y \\
\hline
\end{array} \\
\hline
\end{array}
$$

Quantifier Constructions (Schematic)

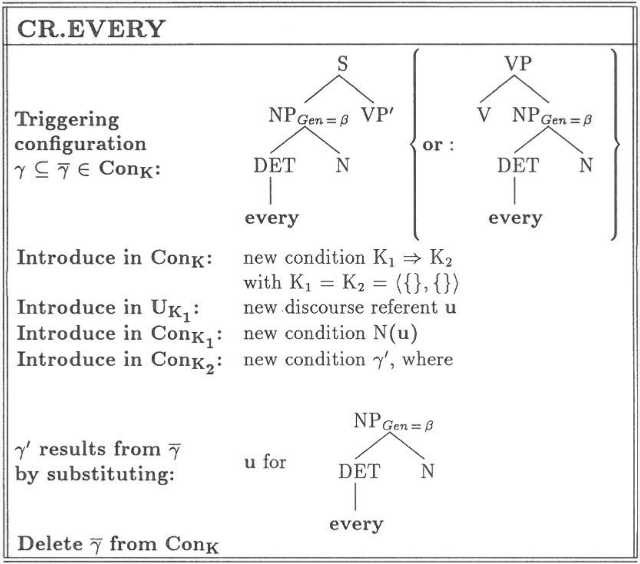


Quantifier Constructions (Schematic)

- $\begin{array}{|c|}\hline \ldots \\ \hline \vdots \\ \text{Every } A\ B\text{s} \\ \hline \end{array} \rightsquigarrow \begin{array}{|c|}\hline \ldots \\ \hline \vdots \\ \begin{array}{|c|}\hline x \\ \hline Ax \\ \hline \end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline Bx \\ \hline \end{array} \\ \hline \end{array}$

- $\begin{array}{|c|}\hline \ldots \\ \hline \vdots \\ \text{No } A\ B\text{s} \\ \hline \end{array} \rightsquigarrow \begin{array}{|c|}\hline \ldots \\ \hline \vdots \\ \neg \begin{array}{|c|}\hline x \\ \hline Ax \\ Bx \\ \hline \end{array} \\ \hline \end{array}$

DRS Construction Algorithm: ‘Every’



CR.EVERY

Triggering configuration $\gamma \subseteq \overline{\gamma} \in \mathbf{Con_K}$**:**

$[_{S}\ [_{NP_{Gen=\beta}}\ [_{DET}\ \textbf{every}]\ [_{N}]]\ [_{VP'}]]$ or : $[_{VP}\ [_{V}]\ [_{NP_{Gen=\beta}}\ [_{DET}\ \textbf{every}]\ [_{N}]]]$

Introduce in $\mathbf{Con_K}$**:** new condition $K_1 \Rightarrow K_2$ with $K_1 = K_2 = \langle\{\},\{\}\rangle$

Introduce in $\mathbf{U_{K_1}}$**:** new discourse referent $\mathbf{u}$

Introduce in $\mathbf{Con_{K_1}}$**:** new condition $N(\mathbf{u})$

Introduce in $\mathbf{Con_{K_2}}$**:** new condition γ', where

γ' **results from** $\overline{\gamma}$ **by substituting:** $\mathbf{u}$ for $[_{NP_{Gen=\beta}}\ [_{DET}\ \textbf{every}]\ [_{N}]]$

Delete $\overline{\gamma}$ **from** $\mathbf{Con_K}$

Event Anaphora

(13) John screamed. That scared Max.

- Include *event indexes* in the referents, and have verb phrases denote events.

$j, m, e^{\text{scream}}, e^{\text{scare}}, e$
John(j)
$\texttt{scream}(e^{\text{scream}}, j)$
Max(m)
$\texttt{scare}(e^{\text{scare}}, e, m)$
$e = e^{\text{scream}}$

Tense

– Add referents for *times* and a special constant *n* (=*now*)

(14) It is sunny now, but it will be raining.

(15) It was sunny.

$$\begin{array}{|c|}\hline e_1, e_2, t_1, t_2 \\ \hline \texttt{sunny}(e_1) \\ at(e_1, t_1) \\ t_1 = n \\ \texttt{raining}(e_2) \\ at(e_2, t_2) \\ n < t_2 \\ \hline \end{array} \qquad \begin{array}{|c|}\hline e_1, t_1 \\ \hline \texttt{sunny}(e_1) \\ at(e_1, t_1) \\ t_1 < n \\ \hline \end{array}$$

DRT

DRT to SDRT

DRT to SDRT

(16) IF a farmer has a donkey, THEN **he** beats **it**.

$$\begin{array}{|c|}\hline \\ \hline \begin{array}{|l|}\hline f, d \\ \hline \texttt{farmer}(\mathrm{f}) \\ \texttt{donkey}(\mathrm{d}) \\ \texttt{own}(\mathrm{d,f}) \\ \hline\end{array} \Rightarrow \begin{array}{|c|}\hline \\ \hline \texttt{beats}(\mathrm{f,d}) \\ \hline\end{array} \\ \hline\end{array}$$

(16) IF a farmer has a donkey, THEN **he** beats **it**.

$$\begin{array}{|c|}\hline \\ \hline \Rightarrow \left(\begin{array}{|l|}\hline f, d \\ \hline \texttt{farmer}(\mathrm{f}) \\ \texttt{donkey}(\mathrm{d}) \\ \texttt{own}(\mathrm{f,d}) \\ \hline \end{array} , \begin{array}{|l|}\hline \\ \hline \texttt{beats}(\mathrm{f,d}) \\ \hline \end{array} \right) \\ \hline \end{array}$$

From DRT to SDRT

(16) IF a farmer has a donkey, THEN **he** beats **it**.

$$\begin{array}{|l|}\hline \pi_1, \pi_2, \pi \\ \hline \\ \pi_1 : \begin{array}{|l|}\hline f, d \\ \hline \texttt{farmer}(f) \\ \texttt{donkey}(d) \\ \texttt{own}(f,d) \\ \hline \end{array} \\ \\ \pi_2 : \begin{array}{|l|}\hline \\ \hline \texttt{beats}(f, d) \\ \hline \end{array} \\ \\ \pi : \Rightarrow (\pi_1, \pi_2) \\ \hline \end{array}$$

From DRT to SDRT

(16) IF a farmer has a donkey, THEN **he** beats **it**.

$$
\begin{array}{|l|}
\hline
\pi, \pi_1, \pi_2 \\
\hline
\pi_1 : \begin{array}{|l|}
\hline
f, d \\
\hline
\texttt{farmer}(f) \\
\texttt{donkey}(d) \\
\texttt{own}(f,d) \\
\hline
\end{array} \\
\pi_2 : \begin{array}{|l|}
\hline
\\
\hline
\texttt{beats}(f,d) \\
\hline
\end{array} \\
\pi : \text{Consequence}(\pi_1, \pi_2) \\
\hline
\end{array}
$$

Segmentation

- What just happened?
- We have taken the units of the DRS and assigned labels to them.
- Then we stated how the labels relate.

SDRSs

- Let DRSs denote the set of well-formed DRSs..
- Let Rel be a set of coherence relations.
- An SDRS is a super-DRS describing the rhetorical structure of many normal DRSs (aka the “microstructure”).

Segmented Discourse Representation Structure

An *SDRS* is a tuple $(\Pi, \mathcal{F}, L)$ with

- Π a set of labels,
- Let REL^{Π} be the set $\{R(\alpha, \beta) \mid R \in \text{Rel} \wedge \alpha, \beta \in \Pi\}$ closed under conjunction.
- $\mathcal{F}$ is a function, $\Pi \rightarrow \text{DRSs} \cup \text{REL}^{\Pi}$.
- $L \in \Pi$ (the *last* label).

Example

(17) IF a farmer has a donkey, THEN **he** beats **it**.

$K = (\Pi, \mathcal{F}, L)$, where:

- $\Pi = \{\pi_1, \pi_2, \pi\}$
- $\mathcal{F} = \left\{ \pi_1 \mapsto \begin{array}{|l|}\hline f, d \\ \hline \texttt{farmer}(f) \\ \texttt{donkey}(d) \\ \texttt{own}(f, d) \\ \hline \end{array}, \begin{array}{l} \pi_2 \mapsto \begin{array}{|c|}\hline \\ \hline \texttt{beats}(f, d) \\ \hline \end{array}, \\ \pi \mapsto \text{Consequence}(\pi_1, \pi_2) \end{array} \right\}$
- $L = \pi_2$

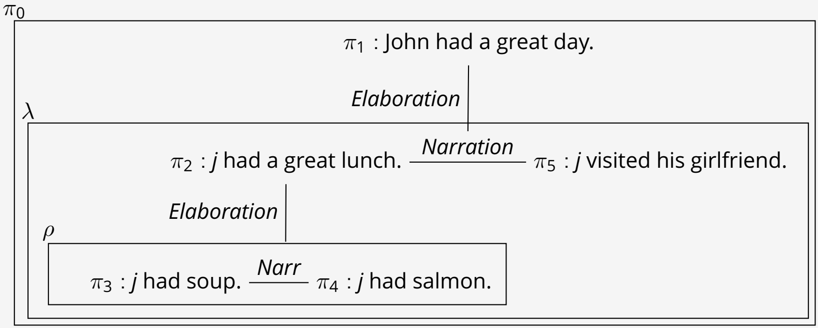
Bigger Example

(18) π_1 : John had a great day.
π_2 : He had a great lunch .
π_3 : He had soup.
π_4 : Then he had salmon.
π_5 : Afterwards, he visited his girlfriend.

- $\Pi = \{\pi_0, \pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \lambda, \rho\}$.
- $L = \pi_5$.
- $\mathcal{F}(\pi_1), ..., \mathcal{F}(\pi_5)$ are (microstructure) DRSs.
- $\mathcal{F}(\lambda) = \mathit{Elaboration}(\pi_2, \rho) \wedge \mathit{Narration}(\pi_2, \pi_4)$
- $\mathcal{F}(\rho) = \mathit{Narration}(\pi_3, \pi_4)$
- $\mathcal{F}(\pi_0) = \mathit{Elaboration}(\pi_1, \lambda)$

Boxes and Boxes

(19) π_1 : John had a great day.
π_2 : He had a great lunch .
π_3 : He had soup.
π_4 : Then he had salmon.
π_5 : Afterwards, he visited his girlfriend.



What about the Right Frontier?

- Some more definitions:

Outscoping

- Note that $\mathcal{F}$ induces an order on Π.
- Say that $\alpha < \beta$ iff α occurs in $\mathcal{F}(\beta)$.
- Let $\prec$ denote the reflexive transitive closure of $<$.
- Call this relation “outscoping”.

Interpretable SDRS

- A SDRS (Π, $\mathcal{F}$, *L*) is well formed if:
- There is a unique outscoping-maximal label in Π (“root”).
- $\prec$ is anti-symmetric (in particular, then, it has no circles)

The Right Frontier (formally)

- Let $(\Pi, \mathcal{F}, L)$ be a well-formed SDRS.

SDRT-Accessibility

Accessibility is defined recursively:

- *L* is accessible.
- If α is accessible and $\alpha \prec \beta$, then β is accessible.
- If (i) α is accessible, and
  (ii) $R(\beta, \alpha)$ occurs in some $\mathcal{F}(\gamma)$, and
  (iii) *R* is subordinating
  then β is accessible.

Summary

- Elementary Discourse Units are Discourse Representation Structures.
- Segmented Discourse Representation Structures are narrative structures on top of these EDUs
- EDUs (microstructure) are constructed by the DRS construction algorithm.
- Within EDUs, anaphora are guided by DRT-accessibility.
- *Across* EDUs, anaphora are guided by the right frontier.
- Next week: *interpretation* of SDRSs and *construction* of SDRSs.