Discourse Representation Structures

Kind of like files with one big filecard.

A farmer owns a donkey

$x\ y$
$\mathbf{farmer}(x)$
$\mathbf{donkey}(y)$
$\mathbf{owns}(x, y)$

Definition of DRS

A discourse representation structure (DRS) is a mental representation built up by the hearer as the discourse unfolds.

A DRS has two parts:

- a universe, containing a set of discourse referents
- a set of conditions. Conditions can be simple, like **farmer**(x), or complex, like $\neg K$ or $K \Rightarrow K'$, where K and K' are both DRSs.

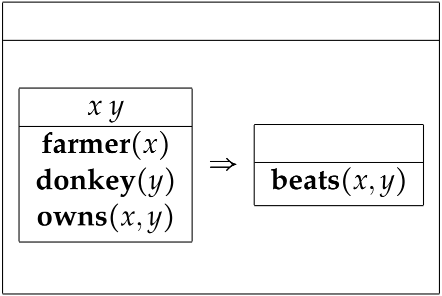
Negation in DRT

Pedro doesn't own a donkey

$$
\begin{array}{|c|}
\hline
x \\
\hline
\\
\textbf{Pedro}(x) \\
\neg \begin{array}{|l|}
\hline
\multicolumn{1}{|c|}{y} \\
\hline
\textbf{donkey}(y) \\
\textbf{owns}(x, y) \\
\hline
\end{array} \\
\hline
\end{array}
$$

Conditionals in DRT

If a farmer owns a donkey, then he beats it



Box vs. linear representation

$x\ y$
$\mathbf{farmer}(x)$
$\mathbf{donkey}(y)$
$\mathbf{owns}(x, y)$

written another way:

$$[x, y : \mathbf{farmer}(x), \mathbf{donkey}(y), \mathbf{owns}(x, y)]$$

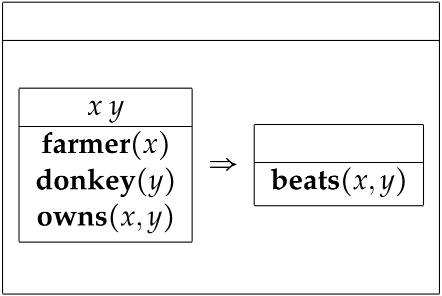
Box vs. linear representation

$$\begin{array}{|c|}\hline x \\ \hline \mathbf{Pedro}(x) \\ \neg \begin{array}{|c|}\hline y \\ \hline \mathbf{donkey}(y) \\ \mathbf{owns}(x,y) \\ \hline \end{array} \\ \hline \end{array}$$

written another way:

$$[x : \mathbf{Pedro}(x), \neg[y : \mathbf{donkey}(y), \mathbf{owns}(x,y)]]$$

Box vs. linear representation



written another way:

$$[: [x, y : \textbf{farmer}(x), \textbf{donkey}(y), \textbf{owns}(x, y)] \Rightarrow [: \textbf{beats}(x, y)]]$$

Truth

Informally, a DRS K is true in a model M if there is a way of associating individuals in the universe of M with the discourse referents of K so that each of the conditions in K is verified in M.

An embedding is a function that maps discourse referents to individuals (like an assignment or sequence). More formally, a DRS is true in a model if there is an embedding that verifies it.

Verifying DRSs with simple conditions

$x\ y$
farmer(x) **donkey**(y) **owns**(x, y)

A function f verifies this DRS with respect to model M if:

- the domain of f contains at least x and y
- according to M it is the case that $f(x)$ is a farmer, $f(y)$ is a donkey, and $f(x)$ chased $f(y)$.

Vad då, "according to M"?

As in predicate logic, we have models $M = \langle D, I \rangle$.

I assigns an extension to every predicate (**farmer**, **donkey**, **owns**, etc.). I(**farmer**) will be a set of individuals; I(**owns**) will be a relation.

So f verifies **farmer**(x) with respect to model $M = \langle D, I \rangle$ if and only if $f(x) \in I$(**farmer**).

Example

$x\ y$
farmer(x) **donkey**(y) **owns**(x, y)

$I(\textbf{Pedro}) = a$
$I(\textbf{farmer}) = \{a, b, c\}$
$I(\textbf{donkey}) = \{d, e, f\}$
$I(\textbf{owns}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$
$I(\textbf{beats}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$

$$g_0 = \varnothing$$

$$g_1 = \begin{bmatrix} x & \rightarrow & a \end{bmatrix}$$

$$g_2 = \begin{bmatrix} x & \rightarrow & b \end{bmatrix}$$

$$g_3 = \begin{bmatrix} x & \rightarrow & b \\ y & \rightarrow & e \end{bmatrix}$$

$$g_4 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & d \end{bmatrix}$$

$$g_5 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & e \end{bmatrix}$$

Verifying negated conditions

x
Pedro(x)
$\neg$ [y | **donkey**(y), **owns**(x, y)]

A function f verifies this DRS if:

- f verifies **Pedro**(x), and
- There is no function g such that: (i) g extends f, and (ii) g verifies $[y : \textbf{donkey}(y), \textbf{owns}(x, y)]$

Extensions of assignments

We say that two functions f and g are compatible if they assign the same values to those arguments for which they are both defined.

I.e., f and g are compatible if for any a which belongs to the domain of both f and g:

$$f(a) = g(a)$$

g is called an extension of f if g is compatible with f and the domain of g includes the domain of f.

Thus if g is an extension of f then f and g assign the same values to all arguments for which f is defined, while g may (though it need not) be defined for some additional arguments as well.

Verification of negated conditions

An embedding function f verifies a condition of the form $\neg K$ with respect to model M iff:

There is no function g such that:

- g extends f
- g verifies K

Example

x
Pedro(x) $\neg$ [y | **donkey**(y), **owns**(x, y)]

$I(\textbf{Pedro}) = a$
$I(\textbf{farmer}) = \{a, b, c\}$
$I(\textbf{donkey}) = \{d, e, f\}$
$I(\textbf{owns}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$
$I(\textbf{beats}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$

$$g_0 = \varnothing$$

$$g_1 = \begin{bmatrix} x & \rightarrow & a \end{bmatrix}$$

$$g_2 = \begin{bmatrix} x & \rightarrow & b \end{bmatrix}$$

$$g_3 = \begin{bmatrix} x & \rightarrow & b \\ y & \rightarrow & e \end{bmatrix}$$

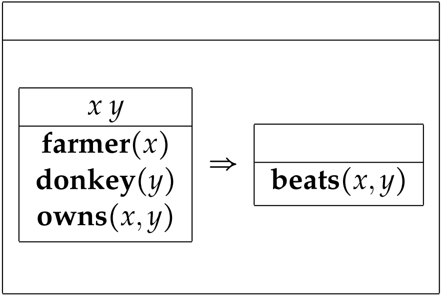
$$g_4 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & d \end{bmatrix}$$

$$g_5 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & e \end{bmatrix}$$

Semantics of conditionals

f verifies a condition of the form $K \Rightarrow K'$ with respect to model M if and only if:

For all extensions g of f that verify K, there is an extension h of g that verifies K'.



$$g_0 = \varnothing$$

$$g_1 = \begin{bmatrix} x & \rightarrow & a \end{bmatrix}$$

$$g_2 = \begin{bmatrix} x & \rightarrow & b \end{bmatrix}$$

$$g_3 = \begin{bmatrix} x & \rightarrow & b \\ y & \rightarrow & e \end{bmatrix}$$

$$g_4 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & d \end{bmatrix}$$

$$g_5 = \begin{bmatrix} x & \rightarrow & a \\ y & \rightarrow & e \end{bmatrix}$$

$I(\textbf{Pedro}) = a$
$I(\textbf{farmer}) = \{a, b, c\}$
$I(\textbf{donkey}) = \{d, e, f\}$
$I(\textbf{owns}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$
$I(\textbf{beats}) = \{\langle a, d\rangle, \langle b, e\rangle, \langle b, f\rangle\}$

Recall: f verifies $K \Rightarrow K'$ iff for all extensions g of f that verify K, there is an extension h of g that verifies K'.

Consequences

Given how truth is defined, it turns out:

- Unembedded indefinites get existential interpretation
- Indefinites acquire universal import in conditionals(!!)
- Indefinites can bind from antecedent to consequent

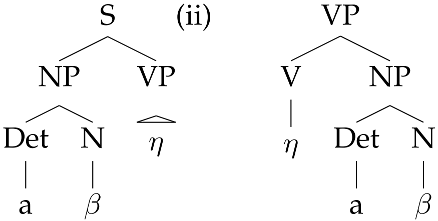
How do we get the DRSs?

- DRS construction rules + construction algorithm.
- This algorithm consists of instructions saying for each expression of a given fragment of natural language how to build or modify the DRS.

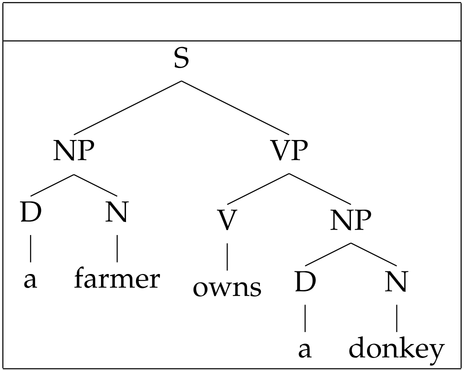
Construction Rule: CR.ID

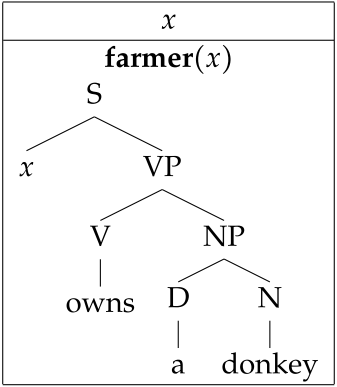
Triggering configurations

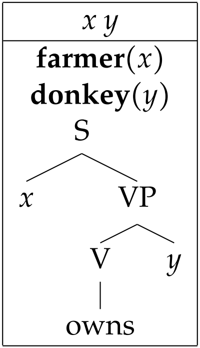
(i) (ii)



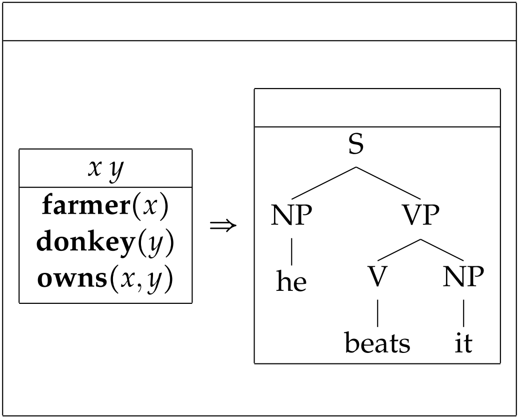
Operations: (i) Introduce new referent **u** in the universe.
(ii) introduce a new condition [N](**u**).
(iii) substitute **u** for the NP.


S
NP
VP
D
N
a
farmer
V
NP
owns
D
N
a
donkey


x
farmer(x)
S
x
VP
V
NP
owns
D
N
a
donkey

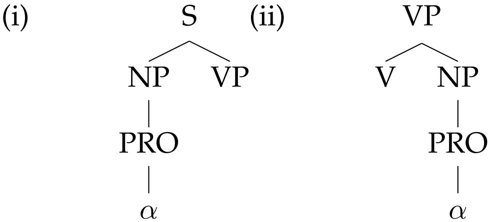

x y
farmer(x)
donkey(y)
S
x
VP
V
y
owns

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

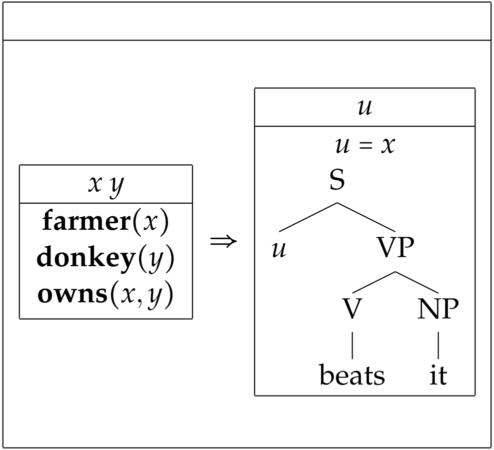

x y
farmer(x)
donkey(y)
owns(x, y)
⇒
S
NP
VP
he
V
NP
beats
it

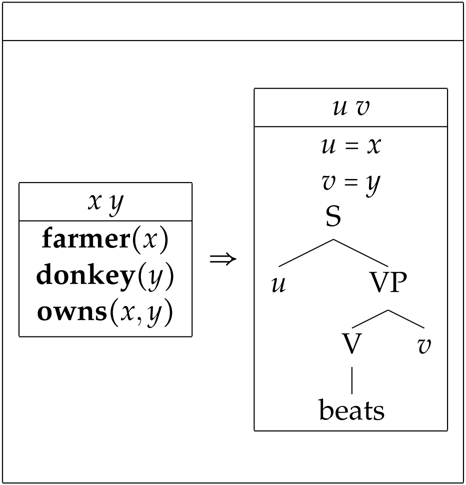
Composition Rule: CR.PRO

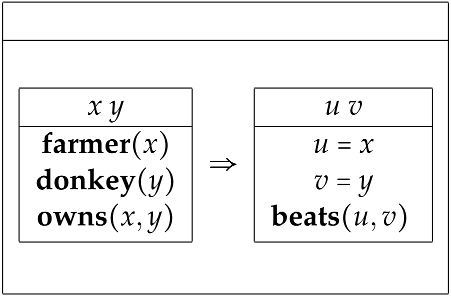
Triggering configurations



Operations: (i) Choose a suitable antecedent **v** such that **v** is accessible. (ii) Introduce a new discourse referent **u** into the universe. (iii) Introduce the condition **u** = **v**. (iv) Substitute **u** for the NP.


x y
farmer(x)
donkey(y)
owns(x, y)
⇒
u
u = x
S
u
VP
V
NP
beats
it


x y
farmer(x)
donkey(y)
owns(x, y)
⇒
u v
u = x
v = y
S
u
VP
V
v
beats


x y
farmer(x)
donkey(y)
owns(x, y)
⇒
u v
u = x
v = y
beats(u, v)

Choosing an antecedent

Recall from CR.PRO: "Choose a suitable antecedent **v** such that **v** is accessible."

What made x a suitable antecedent for u?

- syntactic gender features, which I didn't show

What made x accessible?

- Accessibility is defined in terms of a somewhat complex structural relation among DRSs involving subordination.

Accessibility is the key to "lifespan limitations". Stay tuned!