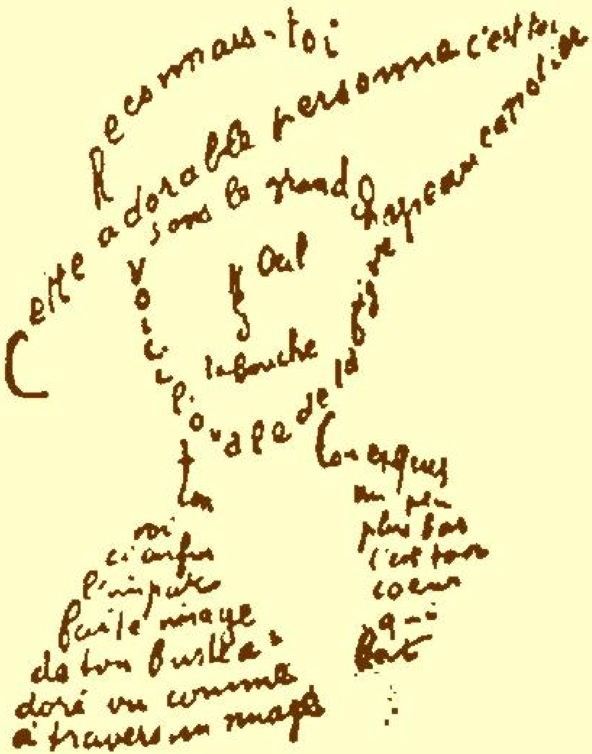

Reconnais-toi
Cette adorable personne c'est toi
sous le grand chapeau canotier
la bouche
voici l'ovale de ta figure
ton cou exquis
un peu plus bas c'est ton cœur qui bat
voici enfin l'imparfaite image de ton buste adoré vu comme à travers un nuage

A Type-Theoretic View of Dynamic Logic

Philippe de Groote

LORIA & Inria-Lorraine

A Type-Theoretic Reconstruction of DRT

A Type-Theoretic Reconstruction of DRT

Motivation:

- to formalize DRT within Church's simple theory of type (aka, Higher-Order Logic), which will allow DRT and Montague semantics to rest on the same logical foundations.

A Type-Theoretic Reconstruction of DRT

Motivation:

- to formalize DRT within Church's simple theory of type (aka, Higher-Order Logic), which will allow DRT and Montague semantics to rest on the same logical foundations.

Challenge:

- to express dynamics using "static" primitives (in particular, to avoid the "destructive assignment" problem, wich necessitates a LISP-like *gensym* operator).

A Type-Theoretic Reconstruction of DRT

Motivation:

- to formalize DRT within Church's simple theory of type (aka, Higher-Order Logic), which will allow DRT and Montague semantics to rest on the same logical foundations.

Challenge:

- to express dynamics using "static" primitives (in particular, to avoid the "destructive assignment" problem, wich necessitates a LISP-like *gensym* operator).

Proposed solution:

- to interpret a sentence according to both its left and right contexts;
- to abstract these two kinds of contexts over the meaning of the sentences.

Typing the left and the right contexts

Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?

Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?

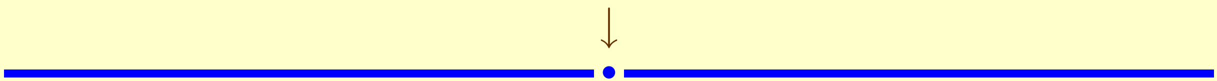
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



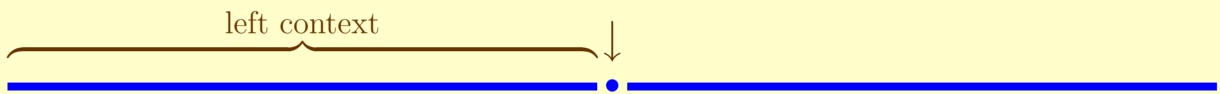
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



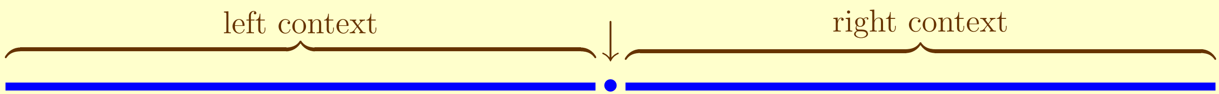
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



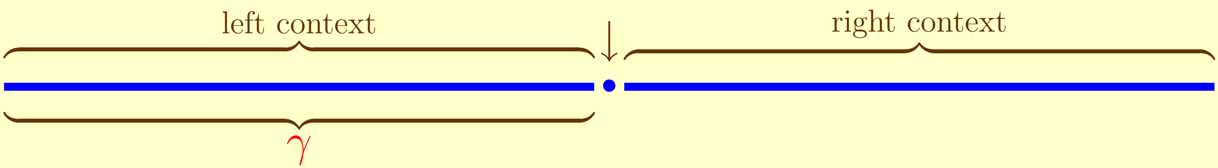
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



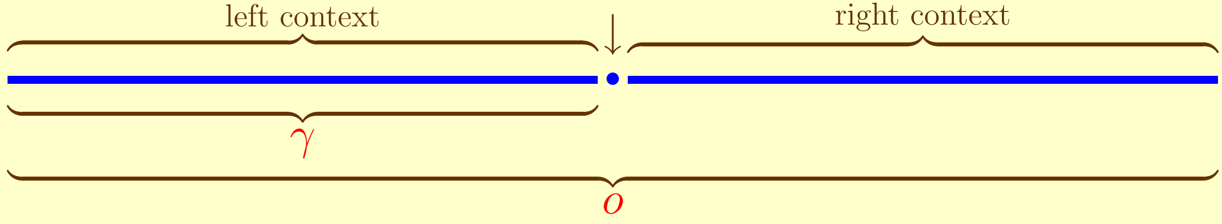
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



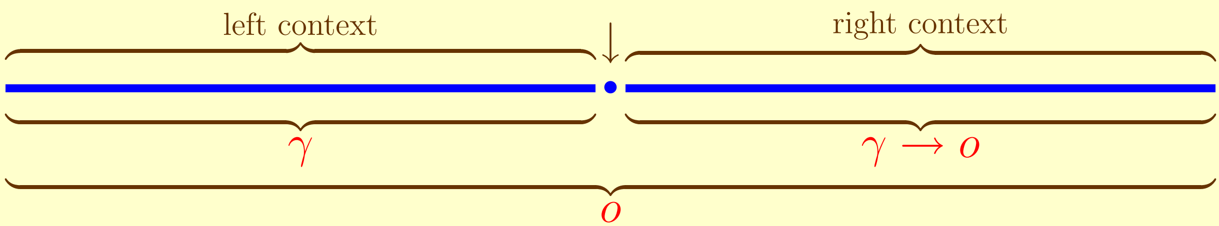
Typing the left and the right contexts

Montague semantics is based on Church's simple type theory, which provides a full hierarchy of functional types built upon two atomic types:

- ι, the type of individuals (a.k.a. entities).
- o, the type of propositions (a.k.a. truth values).

We add a third atomic type, γ, which stands for the type of the left contexts.

What about the type of the right contexts?



Semantic interpretation of the sentences

Semantic interpretation of the sentences

Let s be the syntactic category of sentences. Remember that we intend to abstract our notions of left and right contexts over the meaning of the sentences.

Semantic interpretation of the sentences

Let s be the syntactic category of sentences. Remember that we intend to abstract our notions of left and right contexts over the meaning of the sentences.

$$[\![s]\!] = \gamma \rightarrow (\gamma \rightarrow o) \rightarrow o$$

Semantic interpretation of the sentences

Let s be the syntactic category of sentences. Remember that we intend to abstract our notions of left and right contexts over the meaning of the sentences.

$$[\![s]\!] = \gamma \to (\gamma \to o) \to o$$

Composition of two sentence interpretations

Semantic interpretation of the sentences

Let s be the syntactic category of sentences. Remember that we intend to abstract our notions of left and right contexts over the meaning of the sentences.

$$[\![s]\!] = \gamma \rightarrow (\gamma \rightarrow o) \rightarrow o$$

Composition of two sentence interpretations

$$[\![S_1. \, S_2]\!] = \lambda e \phi. \, [\![S_1]\!] \, e \, (\lambda e'. \, [\![S_2]\!] \, e' \, \phi)$$

Semantic interpretation of the syntactic categories

Semantic interpretation of the syntactic categories

Montague's interpretation

$$
\begin{aligned}
[\![s]\!] &= o \\
[\![n]\!] &= \iota \to o \\
[\![np]\!] &= (\iota \to o) \to o
\end{aligned}
$$

Semantic interpretation of the syntactic categories

Montague's interpretation

$$
\begin{array}{rcl}
[\![s]\!] & = & o \\
[\![n]\!] & = & \iota \to o \\
[\![np]\!] & = & (\iota \to o) \to o
\end{array}
$$

may be rephrased as follows:

$$
\begin{array}{rcll}
[\![s]\!] & = & o & (1) \\
[\![n]\!] & = & \iota \to [\![s]\!] & (2) \\
[\![np]\!] & = & (\iota \to [\![s]\!]) \to [\![s]\!] & (3)
\end{array}
$$

Semantic interpretation of the syntactic categories

Montague's interpretation

$$\begin{aligned} [\![s]\!] &= o \\ [\![n]\!] &= \iota \to o \\ [\![np]\!] &= (\iota \to o) \to o \end{aligned}$$

may be rephrased as follows:

$$\begin{aligned} [\![s]\!] &= o && (1) \\ [\![n]\!] &= \iota \to [\![s]\!] && (2) \\ [\![np]\!] &= (\iota \to [\![s]\!]) \to [\![s]\!] && (3) \end{aligned}$$

Replacing (1) with:

$$[\![s]\!] = \gamma \to (\gamma \to o) \to o$$

Semantic interpretation of the syntactic categories

Montague's interpretation

$$\begin{aligned} [\![s]\!] &= o \\ [\![n]\!] &= \iota \to o \\ [\![np]\!] &= (\iota \to o) \to o \end{aligned}$$

may be rephrased as follows:

$$\begin{aligned} [\![s]\!] &= o && (1) \\ [\![n]\!] &= \iota \to [\![s]\!] && (2) \\ [\![np]\!] &= (\iota \to [\![s]\!]) \to [\![s]\!] && (3) \end{aligned}$$

Replacing (1) with:

$$[\![s]\!] = \gamma \to (\gamma \to o) \to o$$

we obtain:

$$\begin{aligned} [\![n]\!] &= \iota \to \gamma \to (\gamma \to o) \to o \\ [\![np]\!] &= (\iota \to \gamma \to (\gamma \to o) \to o) \to \gamma \to (\gamma \to o) \to o \end{aligned}$$

This interpretation results in handcrafted lexical semantics such as the following:

This interpretation results in handcrafted lexical semantics such as the following:

$$
\begin{aligned}
[\![\text{farmer}]\!] &= \lambda x e \phi.\, \mathbf{farmer}\, x \wedge \phi\, e \\
[\![\text{donkey}]\!] &= \lambda x e \phi.\, \mathbf{donkey}\, x \wedge \phi\, e \\
[\![\text{owns}]\!] &= \lambda o s.\, s\,(\lambda x.\, o\,(\lambda y e \phi.\, \mathbf{own}\, x\, y \wedge \phi\, e)) \\
[\![\text{beats}]\!] &= \lambda o s.\, s\,(\lambda x.\, o\,(\lambda y e \phi.\, \mathbf{beat}\, x\, y \wedge \phi\, e)) \\
[\![\text{who}]\!] &= \lambda r n x e \phi.\, n\, x\, e\,(\lambda e.\, r\,(\lambda \psi.\, \psi\, x)\, e\, \phi) \\
[\![\text{a}]\!] &= \lambda n \psi e \phi.\, \exists x.\, n\, x\, e\,(\lambda e.\, \psi\, x\,(x{::}e)\, \phi) \\
[\![\text{every}]\!] &= \lambda n \psi e \phi.\, (\forall x.\, \neg(n\, x\, e\,(\lambda e.\, \neg(\psi\, x\,(x{::}e)\,(\lambda e.\, \top))))) \wedge \phi\, e \\
[\![\text{it}]\!] &= \lambda \psi e \phi.\, \psi\,(\texttt{sel}\, e)\, e\, \phi
\end{aligned}
$$

This interpretation results in handcrafted lexical semantics such as the following:

$$
\begin{aligned}
[\![\text{farmer}]\!] &= \lambda x e \phi.\, \mathbf{farmer}\, x \wedge \phi\, e \\
[\![\text{donkey}]\!] &= \lambda x e \phi.\, \mathbf{donkey}\, x \wedge \phi\, e \\
[\![\text{owns}]\!] &= \lambda o s.\, s\, (\lambda x.\, o\, (\lambda y e \phi.\, \mathbf{own}\, x\, y \wedge \phi\, e)) \\
[\![\text{beats}]\!] &= \lambda o s.\, s\, (\lambda x.\, o\, (\lambda y e \phi.\, \mathbf{beat}\, x\, y \wedge \phi\, e)) \\
[\![\text{who}]\!] &= \lambda r n x e \phi.\, n\, x\, e\, (\lambda e.\, r\, (\lambda \psi.\, \psi\, x)\, e\, \phi) \\
[\![\text{a}]\!] &= \lambda n \psi e \phi.\, \exists x.\, n\, x\, e\, (\lambda e.\, \psi\, x\, (x{::}e)\, \phi) \\
[\![\text{every}]\!] &= \lambda n \psi e \phi.\, (\forall x.\, \neg(n\, x\, e\, (\lambda e.\, \neg(\psi\, x\, (x{::}e)\, (\lambda e.\, \top))))) \wedge \phi\, e \\
[\![\text{it}]\!] &= \lambda \psi e \phi.\, \psi\, (\texttt{sel}\, e)\, e\, \phi
\end{aligned}
$$

...which might seem a little bit involved.

Questions:

Questions:

- is there a systematic way of obtaining the new lexical semantics from Montague's?

Questions:

- is there a systematic way of obtaining the new lexical semantics from Montague's?
- can we find any "modular" presentation of the approach?

Questions:

- is there a systematic way of obtaining the new lexical semantics from Montague's?
- can we find any "modular" presentation of the approach?
- is there some dynamic logic hidden in the approach?

A Dynamic Logic

A Dynamic Logic

Let $\Omega \triangleq \gamma \rightarrow (\gamma \rightarrow o) \rightarrow o$. We intend to design a logic acting on propositions of type Ω

A Dynamic Logic

Let $\Omega \triangleq \gamma \to (\gamma \to o) \to o$. We intend to design a logic acting on propositions of type Ω

We share with DRT the two following assumptions:

- discourse composition is mainly conjunctive (roughly speaking, a discourse consists in the conjunction of its sentences);
- the main form of quantification is existential (it introduces referential markers).

A Dynamic Logic

Let $\Omega \triangleq \gamma \to (\gamma \to o) \to o$. We intend to design a logic acting on propositions of type Ω

We share with DRT the two following assumptions:

- discourse composition is mainly conjunctive (roughly speaking, a discourse consists in the conjunction of its sentences);
- the main form of quantification is existential (it introduces referential markers).

Consequently, our logic will be based on conjunction and existential quantification (defined as primitives). The other connectives will be obtained using negation (a third primitive) and de Morgan's laws.

Formal Framework

We consider a simply-typed λ-calculus, the terms of which are built upon asignature including the following constants:

Formal Framework

We consider a simply-typed λ-calculus, the terms of which are built upon asignature including the following constants:

FIRST-ORDER LOGIC

$$
\begin{array}{lcll}
\top & : & o & (\textit{truth}) \\
\neg & : & o \rightarrow o & (\textit{negation}) \\
\wedge & : & o \rightarrow o \rightarrow o & (\textit{conjunction}) \\
\exists & : & (\iota \rightarrow o) \rightarrow o & (\textit{existential quantification})
\end{array}
$$

Formal Framework

We consider a simply-typed λ-calculus, the terms of which are built upon asignature including the following constants:

FIRST-ORDER LOGIC

$$\begin{array}{rcll} \top & : & o & (\textit{truth}) \\ \neg & : & o \to o & (\textit{negation}) \\ \wedge & : & o \to o \to o & (\textit{conjunction}) \\ \exists & : & (\iota \to o) \to o & (\textit{existential quantification}) \end{array}$$

DYNAMIC PRIMITIVES

$$\begin{array}{rcll} :: & : & \iota \to \gamma \to \gamma & (\textit{context updating}) \\ \mathsf{sel} & : & \gamma \to \iota & (\textit{choice operator}) \end{array}$$

Conjunction

Conjunction

Conjunction is nothing but sentence composition. We therefore define:

$$A \sqcap B \triangleq \lambda e \phi.\, A\, e\, (\lambda e.\, B\, e\, \phi)$$

Conjunction

Conjunction is nothing but sentence composition. We therefore define:

$$A \sqcap B \triangleq \lambda e\phi.\, A\, e\, (\lambda e.\, B\, e\, \phi)$$

Existential quantification

Conjunction

Conjunction is nothing but sentence composition. We therefore define:

$$A \sqcap B \triangleq \lambda e\phi.\, A\, e\, (\lambda e.\, B\, e\, \phi)$$

Existential quantification

Existential quantification introduces "reference markers". It is therefore responsible for context updating:

$$\Sigma x.\, P\, x \triangleq \lambda e\phi.\, \exists x.\, P\, x\, (x{::}e)\, \phi$$

Conjunction

Conjunction is nothing but sentence composition. We therefore define:

$$A \sqcap B \triangleq \lambda e \phi.\, A\, e\, (\lambda e.\, B\, e\, \phi)$$

Existential quantification

Existential quantification introduces "reference markers". It is therefore responsible for context updating:

$$\Sigma x.\, P\, x \triangleq \lambda e \phi.\, \exists x.\, P\, x\, (x{::}e)\, \phi$$

Negation

Conjunction

Conjunction is nothing but sentence composition. We therefore define:

$$A \sqcap B \triangleq \lambda e \phi.\, A\, e\, (\lambda e.\, B\, e\, \phi)$$

Existential quantification

Existential quantification introduces “reference markers”. It is therefore responsible for context updating:

$$\Sigma x.\, P\, x \triangleq \lambda e \phi.\, \exists x.\, P\, x\, (x{::}e)\, \phi$$

Negation

We do not want the continuation of the discourse to fall into the scope of the negation. Consequently, negation must be defined as follows:

$$\sim A \triangleq \lambda e\phi.\, \neg\,(A\, e\,(\lambda e.\, \top)) \wedge \phi\, e$$

Implication and Universal Quantification

Implication and Universal Quantification

These are defined using de Morgan's laws:

$$
\begin{aligned}
A \sqsupset B &\triangleq \sim(A \sqcap \sim B) \\
\Pi x.\, P\, x &\triangleq \sim\Sigma x.\, \sim(P\, x)
\end{aligned}
$$

Implication and Universal Quantification

These are defined using de Morgan's laws:

$$\begin{aligned} A \sqsupset B &\triangleq {\sim}(A \sqcap {\sim}B) \\ \Pi x.\, P\, x &\triangleq {\sim}\Sigma x.\, {\sim}(P\, x) \end{aligned}$$

Embedding of first-order logic into dynamic logic

Implication and Universal Quantification

These are defined using de Morgan's laws:

$$\begin{aligned} A \sqsupset B &\triangleq {\sim}(A \sqcap {\sim}B) \\ \Pi x.\, P\,x &\triangleq {\sim}\Sigma x.\, {\sim}(P\,x) \end{aligned}$$

Embedding of first-order logic into dynamic logic

$$\begin{aligned} \overline{R\,t_1\,\ldots\,t_n} &= \lambda e\phi.\, R\,t_1\,\ldots\,t_n \wedge \phi\,e \\ \overline{\neg A} &= {\sim}\overline{A} \\ \overline{A \wedge B} &= \overline{A} \sqcap \overline{B} \\ \overline{\exists x.\, A} &= \Sigma x.\, \overline{A} \end{aligned}$$

Implication and Universal Quantification

These are defined using de Morgan's laws:

$$\begin{aligned} A \sqsupset B &\triangleq {\sim}(A \sqcap {\sim}B) \\ \Pi x.\, P\, x &\triangleq {\sim}\Sigma x.\, {\sim}(P\, x) \end{aligned}$$

Embedding of first-order logic into dynamic logic

$$\begin{aligned} \overline{R\, t_1\, \ldots\, t_n} &= \lambda e \phi.\, R\, t_1\, \ldots\, t_n \wedge \phi\, e \\ \overline{\neg A} &= {\sim}\overline{A} \\ \overline{A \wedge B} &= \overline{A} \sqcap \overline{B} \\ \overline{\exists x.\, A} &= \Sigma x.\, \overline{A} \end{aligned}$$

This embedding is such that, for every term e of type γ:

$$A \equiv \overline{A}\, e\, (\lambda e.\, \top)$$

Donkey Sentence Revisited

Donkey Sentence Revisited

Montague-like semantic interpretation:

$$
\begin{aligned}
[\![\text{farmer}]\!] &= \mathbf{farmer} \\
[\![\text{donkey}]\!] &= \mathbf{donkey} \\
[\![\text{owns}]\!] &= \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \mathbf{own}\, x\, y)) \\
[\![\text{beats}]\!] &= \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \mathbf{beat}\, x\, y)) \\
[\![\text{who}]\!] &= \lambda RQx.\, Q\, x \wedge R\,(\lambda P.\, P\, x) \\
[\![\text{a}]\!] &= \lambda PQ.\, \exists x.\, P\, x \wedge Q\, x \\
[\![\text{every}]\!] &= \lambda PQ.\, \forall x.\, P\, x \supset Q\, x \\
[\![\text{it}]\!] &= ???
\end{aligned}
$$

Donkey Sentence Revisited

Montague-like semantic interpretation:

$$
\begin{aligned}
[\![\text{farmer}]\!] &= \mathbf{farmer} \\
[\![\text{donkey}]\!] &= \mathbf{donkey} \\
[\![\text{owns}]\!] &= \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \mathbf{own}\, x\, y)) \\
[\![\text{beats}]\!] &= \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \mathbf{beat}\, x\, y)) \\
[\![\text{who}]\!] &= \lambda RQx.\, Q\, x \wedge R\,(\lambda P.\, P\, x) \\
[\![\text{a}]\!] &= \lambda PQ.\, \exists x.\, P\, x \wedge Q\, x \\
[\![\text{every}]\!] &= \lambda PQ.\, \forall x.\, P\, x \supset Q\, x \\
[\![\text{it}]\!] &= ???
\end{aligned}
$$

Dynamic interpretation:

$$
\begin{array}{rcl}
[\![\text{farmer}]\!] & = & \overline{\textbf{farmer}} \\
[\![\text{donkey}]\!] & = & \overline{\textbf{donkey}} \\
[\![\text{owns}]\!] & = & \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \overline{\textbf{own}}\, x\, y)) \\
[\![\text{beats}]\!] & = & \lambda OS.\, S\,(\lambda x.\, O\,(\lambda y.\, \overline{\textbf{beat}}\, x\, y)) \\
[\![\text{who}]\!] & = & \lambda RQx.\, Q\, x \sqcap R\,(\lambda P.\, P\, x) \\
[\![\text{a}]\!] & = & \lambda PQ.\, \Sigma x.\, P\, x \sqcap Q\, x \\
[\![\text{every}]\!] & = & \lambda PQ.\, \Pi x.\, P\, x \sqsupset Q\, x \\
[\![\text{it}]\!] & = & \lambda Pe\phi.\, P\,(\texttt{sel}\, e)\, e\, \phi
\end{array}
$$

With the dynamic interpretation we have that:

$$[\![\text{beats}]\!]\ [\![\text{it}]\!]\ ([\![\text{every}]\!]\ ([\![\text{who}]\!]\ ([\![\text{owns}]\!]\ ([\![\text{a}]\!]\ [\![\text{donkey}]\!]))\ [\![\text{farmer}]\!]))$$

With the dynamic interpretation we have that:

$$[\![\text{beats}]\!]\,[\![\text{it}]\!]\,([\![\text{every}]\!]\,([\![\text{who}]\!]\,([\![\text{owns}]\!]\,([\![\text{a}]\!]\,[\![\text{donkey}]\!]))\,[\![\text{farmer}]\!]))$$

β-reduces to the following term (modulo de Morgan's laws):

$$\lambda e\phi.\,(\forall x.\,\mathbf{farmer}\,x \supset (\forall y.\,\mathbf{donkey}\,y \supset (\mathbf{own}\,x\,y \supset \mathbf{beat}\,x\,(\mathsf{sel}\,(x::y::e))))) \wedge \phi\,e$$

With the dynamic interpretation we have that:

$$[\![\text{beats}]\!]\,[\![\text{it}]\!]\,([\![\text{every}]\!]\,([\![\text{who}]\!]\,([\![\text{owns}]\!]\,([\![\text{a}]\!]\,[\![\text{donkey}]\!]))\,[\![\text{farmer}]\!]))$$

β-reduces to the following term (modulo de Morgan's laws):

$$\lambda e\phi.\,(\forall x.\,\mathbf{farmer}\,x \supset (\forall y.\,\mathbf{donkey}\,y \supset (\mathbf{own}\,x\,y \supset \mathbf{beat}\,x\,(\mathsf{sel}\,(x{::}y{::}e))))) \wedge \phi\,e$$

that is, assuming that sel is a "perfect" anaphora resolution operator:

$$\lambda e\phi.\,(\forall x.\,\mathbf{farmer}\,x \supset (\forall y.\,\mathbf{donkey}\,y \supset (\mathbf{own}\,x\,y \supset \mathbf{beat}\,x\,y))) \wedge \phi\,e$$

The Higher-Order Case

The Higher-Order Case

Define type "dynamization" as follows:

$$\begin{aligned}\mathsf{D}\iota &= \iota \\ \mathsf{D}o &= \Omega \\ \mathsf{D}(\alpha \to \beta) &= \mathsf{D}\alpha \to \mathsf{D}\beta\end{aligned}$$

The Higher-Order Case

Define type "dynamization" as follows:

$$\begin{aligned} \mathsf{D}\iota &= \iota \\ \mathsf{D}o &= \Omega \\ \mathsf{D}(\alpha \to \beta) &= \mathsf{D}\alpha \to \mathsf{D}\beta \end{aligned}$$

Then, define term "dynamization" as follows:

$$\begin{array}{rcll} \mathsf{D}t &=& \lambda x_1 \ldots x_n.\, t\,(\mathsf{R}_{\mathsf{nil}} x_1)\,\ldots\,(\mathsf{R}_{\mathsf{nil}} x_n) & \text{at type } \alpha_1 \to \ldots \alpha_n \to \iota \\ \mathsf{D}t &=& \lambda x_1 \ldots x_n e \phi.\, t\,(\mathsf{R}_e x_1)\,\ldots\,(\mathsf{R}_e x_n) \wedge (\phi\, e) & \text{at type } \alpha_1 \to \ldots \alpha_n \to o \\ \\ \mathsf{R}_e t &=& \lambda x_1 \ldots x_n.\, t\,(\mathsf{D}x_1)\,\ldots\,(\mathsf{D}x_n) & \text{at type } \mathsf{D}(\alpha_1 \to \ldots \alpha_n \to \iota) \\ \mathsf{R}_e t &=& \lambda x_1 \ldots x_n.\, t\,(\mathsf{D}x_1)\,\ldots\,(\mathsf{D}x_n)\, e\,(\lambda e.\, \top) & \text{at type } \mathsf{D}(\alpha_1 \to \ldots \alpha_n \to o) \end{array}$$

Finally, define λ-term translation as follows:

$$
\begin{array}{rcl}
\overline{x} & = & x \\
\overline{\wedge} & = & \sqcap \\
\overline{\exists} & = & \Sigma \\
\overline{\neg} & = & \sim \\
\overline{k} & = & \mathsf{D}k \quad \text{for the other constants} \\
\overline{\lambda x.\, t} & = & \lambda x.\, \overline{t} \\
\overline{t\, u} & = & \overline{t}\, \overline{u}
\end{array}
$$

Finally, define λ-term translation as follows:

$$
\begin{array}{rcll}
\overline{x} & = & x & \\
\overline{\wedge} & = & \sqcap & \\
\overline{\exists} & = & \Sigma & \\
\overline{\neg} & = & \sim & \\
\overline{k} & = & \mathsf{D}k & \text{for the other constants} \\
\overline{\lambda x.\, t} & = & \lambda x.\, \overline{t} & \\
\overline{t\, u} & = & \overline{t}\, \overline{u} &
\end{array}
$$

Then, for every closed term t of type o, and every context e, we have that:

$$\overline{t}\, e\, (\lambda e.\, \top) \equiv t$$

Comparison with existing works

Comparison with existing works

Most existing works on dynamics (DRT, Muskens', Groenendijk & Stokhof's) interpret dynamic propositions as binary relations on states (a.k.a., assignments or environments).

Comparison with existing works

Most existing works on dynamics (DRT, Muskens', Groenendijk & Stokhof's) interpret dynamic propositions as binary relations on states (a.k.a., assignments or environments).

In our setting, these would be terms of type:

$$\gamma \to \gamma \to o,$$

Comparison with existing works

Most existing works on dynamics (DRT, Muskens', Groenendijk & Stokhof's) interpret dynamic propositions as binary relations on states (a.k.a., assignments or environments).

In our setting, these would be terms of type:

$$\gamma \to \gamma \to o,$$

and the semantics of Groenendijk & Stokhof's DPL would be rephrased as follows:

$$\begin{array}{rcll}
A_{\mathrm{d}} & \triangleq & \lambda gh.\, h{=}g \wedge A & \text{(atomic proposition)} \\
(\neg P)_{\mathrm{d}} & \triangleq & \lambda gh.\, h{=}g \wedge \neg(\exists k.\, P_{\mathrm{d}}\, h\, k) & \text{(negation)} \\
(P \wedge Q)_{\mathrm{d}} & \triangleq & \lambda gh.\, \exists k.\, P_{\mathrm{d}}\, g\, k \wedge Q_{\mathrm{d}}\, k\, h & \text{(conjunction)} \\
(\exists x.\, P)_{\mathrm{d}} & \triangleq & \lambda gh.\, \exists k.\, k[x]g \wedge P_{\mathrm{d}}\, k\, h & \text{(existential)}
\end{array}$$

There exists a canonical embedding $\lceil \cdot \rceil$ from $\gamma \to \gamma \to o$ into $\gamma \to (\gamma \to o) \to o$:

$$\lceil R \rceil \triangleq \lambda e \phi.\, \exists e'.\, \phi\, e' \wedge R\, e\, e'$$

There exists a canonical embedding $\lceil\cdot\rceil$ from $\gamma \to \gamma \to o$ into $\gamma \to (\gamma \to o) \to o$:

$$\lceil R \rceil \triangleq \lambda e\phi.\, \exists e'.\, \phi\, e' \wedge R\, e\, e'$$

Then, we have:

$$\begin{aligned}
\lceil A_{\mathrm{d}} \rceil &\equiv \overline{A} \\
\lceil (\neg P)_{\mathrm{d}} \rceil &\equiv \sim \lceil P_{\mathrm{d}} \rceil \\
\lceil (P \wedge Q)_{\mathrm{d}} \rceil &\equiv \lceil P_{\mathrm{d}} \rceil \sqcap \lceil Q_{\mathrm{d}} \rceil
\end{aligned}$$

There exists a canonical embedding $\lceil\cdot\rceil$ from $\gamma \to \gamma \to o$ into $\gamma \to (\gamma \to o) \to o$:

$$\lceil R \rceil \triangleq \lambda e\phi.\, \exists e'.\, \phi\, e' \wedge R\, e\, e'$$

Then, we have:

$$\begin{aligned} \lceil A_{\mathrm{d}} \rceil &\equiv \overline{A} \\ \lceil (\neg P)_{\mathrm{d}} \rceil &\equiv \sim \lceil P_{\mathrm{d}} \rceil \\ \lceil (P \wedge Q)_{\mathrm{d}} \rceil &\equiv \lceil P_{\mathrm{d}} \rceil \sqcap \lceil Q_{\mathrm{d}} \rceil \end{aligned}$$

As for the existential quantifier:

$$\begin{aligned} \lceil (\exists x.\, P)_{\mathrm{d}} \rceil &= \lambda e\phi.\, \exists e'.\, \phi\, e' \wedge \exists k.\, k[x]e \wedge \lceil P_{\mathrm{d}} \rceil\, k\, e' \\ \Sigma x.\, \lceil P_{\mathrm{d}} \rceil &= \lambda e\phi.\, \exists e'.\, \phi\, e' \wedge \exists x.\, \lceil P_{\mathrm{d}} \rceil\, (x :: e)\, e' \end{aligned}$$

Conclusions

Conclusions

What do we gain?

Conclusions

What do we gain?

- No destructive assignment.

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.
- Relatively independent of the underlying logic:

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.
- Relatively independent of the underlying logic:
  - generalizes to H.O.L.;

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.
- Relatively independent of the underlying logic:
  - generalizes to H.O.L.;
  - an intuitionistic version could be worked out.

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.
- Relatively independent of the underlying logic:
  - generalizes to H.O.L.;
  - an intuitionistic version could be worked out.
- Relies on well-established existing theories (proof-theory and model theory for free).

Conclusions

What do we gain?

- No destructive assignment.
- Parametric in γ.
- Relatively independent of the underlying logic:
  - generalizes to H.O.L.;
  - an intuitionistic version could be worked out.
- Relies on well-established existing theories (proof-theory and model theory for free).
- Deduction is replaced by computation.