

What syntax doesn't feed semantics

Fake indexicals as indexicals

Emar Maier

ILLC/University of Amsterdam

www.ru.nl/ncs/~emar

ESSLLI Workshop 'What syntax feeds semantics?', Hamburg
August 11, 2008

Introduction: /

- Kaplan (1977,1989):
 - / is indexical, like *today*
 1. context-dependent
 2. directly referential
 - 2D semantics

I am speaking \neq the speaker is speaking

- Heim (1991,2008), Kratzer (1998,2008), Jacobson (2008):
 - / is a pronoun, like *he*
 - pronouns have bound and referential readings

Only I did my homework

sloppy others didn't do theirs: $\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, x)]$

- today: defend Kaplan

Introduction: /

- Kaplan (1977,1989):
 - / is indexical, like *today*
 1. context-dependent
 2. directly referential
 - 2D semantics

I am speaking \neq the speaker is speaking

- Heim (1991,2008), Kratzer (1998,2008), Jacobson (2008):
 - / is a pronoun, like *he*
 - pronouns have bound and referential readings

Only I did my homework

sloppy others didn't do theirs: $\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, x)]$

- today: defend Kaplan

Introduction: /

- Kaplan (1977,1989):
 - / is indexical, like *today*
 1. context-dependent
 2. directly referential
 - 2D semantics

I am speaking \neq the speaker is speaking

- Heim (1991,2008), Kratzer (1998,2008), Jacobson (2008):
 - / is a pronoun, like *he*
 - pronouns have bound and referential readings

Only I did my homework

sloppy others didn't do theirs: $\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, x)]$

- today: defend Kaplan

Introduction: /

- Kaplan (1977,1989):
 - / is indexical, like *today*
 1. context-dependent
 2. directly referential
 - 2D semantics

I am speaking \neq the speaker is speaking

- Heim (1991,2008), Kratzer (1998,2008), Jacobson (2008):
 - / is a pronoun, like *he*
 - pronouns have bound and referential readings

Only I did **my** homework

sloppy others didn't do theirs: $\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, x)]$

- today: defend Kaplan

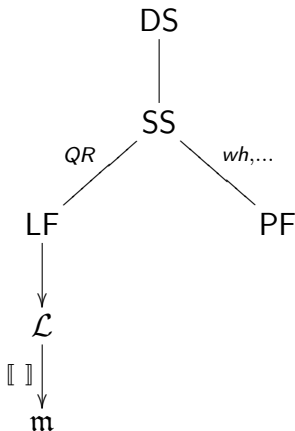
Outline

- 1 Introduction
- 2 Fake indexicals
 - Pronouns in Generative Linguistics
 - VP ellipsis
 - Only
- 3 ... as indexicals
 - Ellipsis resolution by unification
 - Only by unification
 - Avoiding sloppy names
- 4 *De se* binding and *de re* acquaintance
 - Avoiding *de se* names

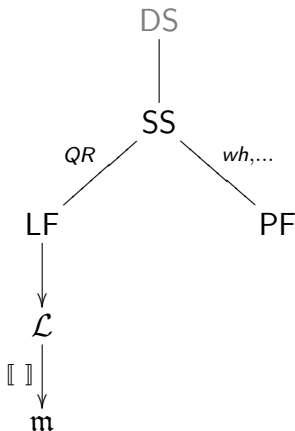
Outline

- 1 Introduction
- 2 Fake indexicals
 - Pronouns in Generative Linguistics
 - VP ellipsis
 - Only
- 3 ... as indexicals
 - Ellipsis resolution by unification
 - Only by unification
 - Avoiding sloppy names
- 4 *De se* binding and *de re* acquaintance
 - Avoiding *de se* names

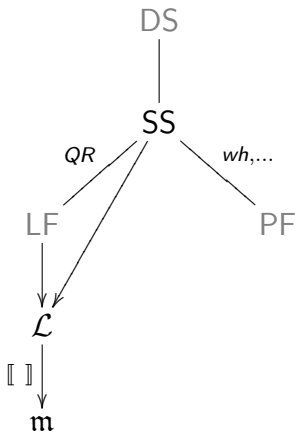
The syntax-semantics interface



The syntax-semantics interface



The syntax-semantics interface



Binding and coreference

SS: John did his homework



Binding and coreference

SS: John did his homework

LF: [John did his homework]

Binding and coreference

SS: John did his homework

LF: [John did his homework]

↓
 \mathcal{L} : do.homework.of(j, x)

Binding and coreference

SS: John did his homework

LF: [John did his homework]

↓
 \mathcal{L} : do.homework.of(j, x)
↓
 m : $\llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of $f(x)$

Binding and coreference

SS: John did his homework

LF: [John did his homework]

↓
 \mathcal{L} : do.homework.of(j, x)
↓
 m : $\llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of $f(x)$

context: his = $f(x)$ = John

Binding and coreference

SS: John did his homework

LF: [John did his homework]

John¹ [t₁ did his₁ homework]

↓
 \mathcal{L} : do.homework.of(j, x)
↓
 m : $\llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of John

context: his = $f(x)$ = John

Binding and coreference

SS: John did his homework

LF: [John did his homework]

John¹ [t₁ did his₁ homework]

↓
 \mathcal{L} : do.homework.of(j, x)

↓
 $\lambda x[\text{do.homework.of}(x, x)](j)$

↓
 $m: \llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of John

context: his = $f(x) = \text{John}$

Binding and coreference

SS: John did his homework

LF: [John did his homework]

John¹ [t₁ did his₁ homework]

↓
 \mathcal{L} : do.homework.of(j, x)

↓
do.homework.of(j, j)

↓
 m : $\llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of John

context: his = $f(x)$ = John

Binding and coreference

SS: John did his homework

LF: [John did his homework]

↓
 \mathcal{L} : do.homework.of(j, x)
↓
 m : $\llbracket \text{do.homework.of}(j, x) \rrbracket_w^f = 1$
iff John did homework of John

context: his = $f(x)$ = John

John¹ [t₁ did his₁ homework]

↓
do.homework.of(j, j)
↓
 $\llbracket \text{do.homework.of}(j, j) \rrbracket_w^f = 1$ iff
John did homework of John

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't like his dad

(context: his = his = John)

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

(context: his = his = John)

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

(context: his = his = John)

$\text{like.dad}(j, x) \wedge \text{like.dad}(p, y)$

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

(context: $f(x)=f(y)=\llbracket j \rrbracket$)
 $\text{like.dad}(j, x) \wedge \text{like.dad}(p, y)$

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

John¹ [t₁ likes his₁ dad] but
Peter¹ [t₁ doesn't like his₁ dad]

(context: $f(x)=f(y)=\llbracket j \rrbracket$)
like.dad(j, x) \wedge like.dad(p, y)

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

John¹ [t₁ likes his₁ dad] but
Peter¹ [t₁ doesn't ~~like his₁ dad~~]

(context: $f(x)=f(y)=\llbracket j \rrbracket$)
like.dad(j, x) \wedge like.dad(p, y)

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
 Peter doesn't ~~like his dad~~

(context: $f(x)=f(y)=\llbracket j \rrbracket$)
 $\text{like.dad}(j, x) \wedge \text{like.dad}(p, y)$

$\text{John}^1 [\text{t}_1 \text{ likes his}_1 \text{ dad}]$ but
 $\text{Peter}^1 [\text{t}_1 \text{ doesn't like his}_1 \text{ dad}]$

$\lambda x[\text{like.dad}(x, x)](j) \wedge$
 $\neg \lambda x[\text{like.dad}(x, x)](p)$

VP ellipsis

PF: John likes his dad but Peter doesn't

- ambiguous:
 - strict: Peter doesn't like John's
 - sloppy: Peter doesn't like his own
- Sag/Williams: reduce to referential-bound ambiguity
 - delete an LF constituent at PF if it's *semantically equivalent* to an earlier constituent at LF

LF: John likes his dad but
Peter doesn't ~~like his dad~~

(context: $f(x)=f(y)=\llbracket j \rrbracket$)
 $\text{like.dad}(j, x) \wedge \text{like.dad}(p, y)$

John¹ [₁ likes his₁ dad] but
Peter¹ [₁ doesn't ~~like his₁ dad~~]

$\text{like.dad}(j, j) \wedge$
 $\neg \text{like.dad}(p, p)$

Binding an indexical?

PF: I like my job, but Sue doesn't



Binding an indexical?

PF: I like my job, but Sue doesn't

LF: I like my job but
Sue doesn't ~~like my job~~

Binding an indexical?

PF: I like my job, but Sue doesn't

LF: I like my job but
Sue doesn't ~~like my job~~

$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$

Binding an indexical?

PF: I like my job, but Sue doesn't

LF: I like my job but
Sue doesn't like ~~my job~~

$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$

$I^1 [t_1 \text{ like my}_1 \text{ job}]$ but
 $Sue^1 [t_1 \text{ doesn't like my}_1 \text{ job}]$

Binding an indexical?

PF: I like my job, but Sue doesn't

LF: I like my job but
 Sue doesn't like my job

$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$

$I^1 [t_1 \text{ like my}_1 \text{ job}] \text{ but}$
 $Sue^1 [t_1 \text{ doesn't like my}_1 \text{ job}]$

$\lambda x[\text{like.job}(x, x)](i) \wedge$
 $\neg \lambda x[\text{like.job}(x, x)](s)$

Binding an indexical?

PF: I like my job, but Sue doesn't

LF: I like my job but
Sue doesn't ~~like my job~~


$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$

$I^1 [t_1 \text{ like my}_1 \text{ job}]$ but
 $Sue^1 [t_1 \text{ doesn't } \text{like my}_1 \text{ job}]$

$\text{like.job}(i, i) \wedge$
 $\neg \text{like.job}(s, s)$

Only

PF: Only I did my homework



Only

PF: Only I did my homework

[Only I]¹ [t₁ did my homework]

Only

PF: Only I did my homework

[Only I]¹ [t₁ did my homework]

$\text{only}(i)(\lambda x[\text{do.hw}(x, i)])$

Only

PF: Only I did my homework

[Only I]¹ [t₁ did my homework]

$\text{only}(i)(\lambda x[\text{do.hw}(x, i)])$

$\text{only}(x)(P) \equiv \forall y[y \neq x \rightarrow \neg P(y)]$

Only

PF: Only I did my homework

[Only I]¹ [t₁ did my homework]

[Only I]¹ [t₁ did my₁ homework]

$\text{only}(i)(\lambda x[\text{do.hw}(x, i)])$

$\text{only}(x)(P) \equiv \forall y[y \neq x \rightarrow \neg P(y)]$

Only

PF: Only I did my homework

[Only I]¹ [t₁ did my homework]

$\text{only}(i)(\lambda x[\text{do.hw}(x, i)])$

[Only I]¹ [t₁ did my₁ homework]

$\text{only}(i)(\lambda x[\text{do.hw}(x, x)])$

$\text{only}(x)(P) \equiv \forall y[y \neq x \rightarrow \neg P(y)]$

Outline

- 1 Introduction
- 2 Fake indexicals
 - Pronouns in Generative Linguistics
 - VP ellipsis
 - Only
- 3 ... as indexicals
 - Ellipsis resolution by unification
 - Only by unification
 - Avoiding sloppy names
- 4 *De se* binding and *de re* acquaintance
 - Avoiding *de se* names

Ellipsis by unification

SS: I like my job, but Sue doesn't

Ellipsis by unification

SS: I like my job, but Sue doesn't

$\text{like.job}(i, i) \wedge \neg P(s)$

Ellipsis by unification

SS: I like my job, but Sue doesn't

$\text{like.job}(i, i) \wedge \neg P(s)$

Ellipsis by unification

SS: I like my job, but Sue doesn't

$$\text{like.job}(\mathbf{i}, i) \wedge \neg P(\mathbf{s})$$
$$P(i) \doteq \text{like.job}(i, i)$$

Ellipsis by unification

SS: I like my job, but Sue doesn't

$\text{like.job}(i, i) \wedge \neg P(s)$

$P(i) \doteq \text{like.job}(i, i)$

$P \mapsto \lambda x[\text{like.job}(x, i)]$

$P \mapsto \lambda x[\text{like.job}(x, x)]$

Ellipsis by unification

SS: I like my job, but Sue doesn't

$$\text{like.job}(i, i) \wedge \neg P(s)$$

$$P(i) \doteq \text{like.job}(i, i)$$

$$P \mapsto \lambda x[\text{like.job}(x, i)]$$

$$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$$

$$P \mapsto \lambda x[\text{like.job}(x, x)]$$

Ellipsis by unification

SS: I like my job, but Sue doesn't

$$\text{like.job}(i, i) \wedge \neg P(s)$$

$$P(i) \doteq \text{like.job}(i, i)$$

$P \mapsto \lambda x [\text{like.job}(x, i)]$	\swarrow	$P \mapsto \lambda x [\text{like.job}(x, x)]$
$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, i)$		$\text{like.job}(i, i) \wedge \neg \text{like.job}(s, s)$

cf. Dalrymple et al. 1991

Only by unification

SS: Only $[I]_F$ did my homework

Only by unification

SS: Only $[I]_F$ did my homework

$$\forall x[x \neq i \rightarrow \neg B(x)]$$

Only by unification

SS: Only $[I]_F$ did my homework

$$\forall x[x \neq i \rightarrow \neg B(x)]$$
$$B(i) \doteq \text{do.hw}(i, i)$$

Only by unification

SS: Only $[I]_F$ did my homework

$$\forall x[x \neq i \rightarrow \neg B(x)]$$

$$B(i) \doteq \text{do.hw}(i, i)$$

$$B \mapsto \lambda x[\text{do.hw}(x, i)]$$

$$B \mapsto \lambda x[\text{do.hw}(x, x)]$$

Only by unification

SS: Only $[I]_F$ did my homework

$$\forall x[x \neq i \rightarrow \neg B(x)]$$

$$B(i) \doteq \text{do.hw}(i, i)$$



$$B \mapsto \lambda x[\text{do.hw}(x, i)]$$

$$\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, i)]$$

$$B \mapsto \lambda x[\text{do.hw}(x, x)]$$

$$\forall x[x \neq i \rightarrow \neg \text{do.hw}(x, x)]$$

cf. Pulman (1997)

Conclusions

- semantic/pragmatic alternative:
 - minimized syntactic levels
 - / is true indexical, interpreted in situ
 - derive strict/sloppy by HOU

Conclusions

- semantic/pragmatic alternative:
 - minimized syntactic levels
 - *I* is true indexical, interpreted in situ
 - derive strict/sloppy by HOU
- Kaplan saved?

No sloppy names

John likes John's job but Sue doesn't

- strict: Sue doesn't like John's job

Only Mary likes Mary's job

- strict: \rightsquigarrow others don't like Mary's

Predictions

- generative:
 - names \neq pronouns
 - Principle C prohibits bound names

Predictions

- generative:
 - names \neq pronouns
 - Principle C prohibits bound names
 - prediction: only reference, only strict

Predictions

- generative:
 - names \neq pronouns
 - Principle C prohibits bound names
 - prediction: only reference, only strict
- pragmatic:
 - names \approx indexicals: directly referential

Predictions

- generative:
 - names \neq pronouns
 - Principle C prohibits bound names
 - prediction: only reference, only strict
- pragmatic:
 - names \approx indexicals: directly referential
 - prediction: strict + sloppy (by HOU)

Pragmatic blocking

- competing alternatives:

(1) Only Mary likes Mary's job

(2) Only Mary likes her job

Pragmatic blocking

- competing alternatives:

(1) Only Mary likes Mary's job

(2) Only Mary likes her job

- (1) violates Principle C
- (1) more marked by referential hierarchy:
 - definite descriptions > names > pronouns

Pragmatic blocking

- competing alternatives:

(1) Only Mary likes Mary's job

(2) Only Mary likes her job

- (1) violates Principle C
- (1) more marked by referential hierarchy:
 - definite descriptions > names > pronouns
- ulterior pragmatic motive for using (1)?

Pragmatic blocking

- competing alternatives:

(1) Only Mary likes Mary's job

(2) Only Mary likes her job

- (1) violates Principle C
- (1) more marked by referential hierarchy:
 - definite descriptions > names > pronouns
- ulterior pragmatic motive for using (1)?
 - topicalizes/presupposes/makes salient Mary's job
 - prioritize background containing Mary's job

Pragmatic blocking

- competing alternatives:

(1) Only Mary likes Mary's job

(2) Only Mary likes her job

- (1) violates Principle C
- (1) more marked by referential hierarchy:
 - definite descriptions > names > pronouns
- ulterior pragmatic motive for using (1)?
 - topicalizes/presupposes/makes salient Mary's job
 - prioritize background containing Mary's job
 - $B \mapsto \{\lambda x[\text{like.job}(x, m)], \lambda x[\text{like.job}(x, x)]\}$

Conclusions

- generative
 - syntax/semantics: PF, LF, SS, \mathcal{L} , m
 - pronouns vs names
 - binding/reference ambiguity: he, she, they, I, you, ...
 - reference: John, Sue, ...
 - in ellipsis, focus, only:
 - reference \rightarrow strict
 - binding \rightarrow sloppy

Conclusions

- generative
 - syntax/semantics: PF, LF, SS, \mathcal{L} , m
 - pronouns vs names
 - binding/reference ambiguity: he, she, they, I, you, ...
 - reference: John, Sue, ...
 - in ellipsis, focus, only:
 - reference \rightarrow strict
 - binding \rightarrow sloppy
- pragmatic
 - semantics/pragmatics: SS, \mathcal{L} , m
 - anaphoric vs directly referential
 - anaphoric: he, she, they, ...
 - referential: I, you, John, today, ...
 - HOU pragmatically derives strict/sloppy
 - sloppy names pragmatically blocked by anaphoric alternative

Outline

- 1 Introduction
- 2 Fake indexicals
 - Pronouns in Generative Linguistics
 - VP ellipsis
 - Only
- 3 ... as indexicals
 - Ellipsis resolution by unification
 - Only by unification
 - Avoiding sloppy names
- 4 *De se* binding and *de re* acquaintance
 - Avoiding *de se* names

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i[\text{safe}(i)]$

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i[\text{safe}(i)]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i[\text{safe}(i)]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$BEL_i[\text{remarkably.calm}(i)]$

De se and de re

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i^* \lambda x[\text{safe}(x)]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$BEL_i[\text{remarkably.calm}(i)]$

De se and *de re*

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i^* \lambda x[\text{safe}(x)]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$BEL_i^{e \times et} \langle i, \lambda x[r.\text{calm}(x)] \rangle$

De se and de re

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$BEL_i^* \lambda x[\text{safe}(x)]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$R(i, i) \wedge BEL_i^* \lambda x[r.\text{calm}(\gamma y[R(x, y)])]$

De se and de re

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

“I'm at safe distance from fire”

$BEL_i^* \lambda x [\text{safe}(x)]$

?I thought that I was remarkably calm

“that guy is remarkably calm”

$R(i, i) \wedge BEL_i^* \lambda x [r.\text{calm}(\gamma y [R(x, y)])]$

$R = \lambda x \lambda y [\text{see.on.tv}(x, y)]$

De se and de re

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought I was at a safe distance from the fire

"I'm at safe distance from fire"

$$R(i, i) \wedge BEL_i^* \lambda x [\text{safe}(\gamma y [R(x, y)])]$$

$$R = \lambda x \lambda y [x = y]$$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$$R(i, i) \wedge BEL_i^* \lambda x [r.\text{calm}(\gamma y [R(x, y)])]$$

$$R = \lambda x \lambda y [\text{see.on.tv}(x, y)]$$

De se and de re

Kaplan is telling the story of the time he didn't realize his pants were on fire while seeing himself on fire on live TV

I thought **I** was at a safe distance from the fire

"I'm at safe distance from fire"

$R(i, i) \wedge BEL_i^* \lambda x [\text{safe}(\gamma y [R(x, y)])]$

$R = \lambda x \lambda y [x = y]$

?I thought that I was remarkably calm

"that guy is remarkably calm"

$R(i, i) \wedge BEL_i^* \lambda x [r.\text{calm}(\gamma y [R(x, y)])]$

$R = \lambda x \lambda y [\text{see.on.tv}(x, y)]$

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

- Chierchia'89: Principle C blocks binding but i_2 - i_1 coreference \Rightarrow *de re* (non-*de se*)

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

- Chierchia'89: Principle C blocks binding but i_2 - i_1 coreference \Rightarrow *de re* (non-*de se*)
- cheaper alternative:

2 Kaplan thought he was at a safe distance from the fire

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

- Chierchia'89: Principle C blocks binding but i_2-i_1 coreference \Rightarrow *de re* (non-*de se*)
- cheaper alternative:

2 Kaplan thought he was at a safe distance from the fire

- ulterior pragmatic motive for using (1)?

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

- Chierchia'89: Principle C blocks binding but i_2-i coreference \Rightarrow *de re* (non-*de se*)
- cheaper alternative:

2 Kaplan thought he was at a safe distance from the fire

- ulterior pragmatic motive for using (1)?
 - Kaplan \in reported thought

De se names

1 # Kaplan thought Kaplan was at a safe distance from the fire

- Chierchia'89: Principle C blocks binding but i_2 - i coreference \Rightarrow *de re* (non-*de se*)
- cheaper alternative:

2 Kaplan thought he was at a safe distance from the fire

- ulterior pragmatic motive for using (1)?
 - Kaplan \in reported thought
- generalization: use marked coref res X only if X matches the reported thought character

Kaplan thought Kaplan was remarkably calm

Kaplan thought the guy on TV was remarkably calm