Bare Plurals and Specificity

Thomas Ede Zimmermann

Frankfurt (Main)

1. Transparency and Specificity
2. Opacity and Monotonicity
3. Unspecific vs. underspecific readings
4. Bare Plurals and Belladonnas

1. Transparency and Specificity

Background assumptions

(i) Ordinary predicates express relations between individuals.

(ii) Singular indefinites express existential quantification.

Three inference schemata

- *Existential Impact*
- *Extensionality*
- *Specificity*

Existential Impact

From

x Rs an N.

infer:

There is at least one N.

Extensionality

From

x Rs an N.

Every N is an M.

Every M is an N.

infer:

x Rs an M.

Specificity

From

x Rs an N.

infer:

Some (specific) individual is Red by x.

Specificity

From

x Rs an N.

infer:

Some (specific) individual is Red by x.

2. Transparency and Specificity

***R* is <u>opaque</u>**

if the above inference schema is invalid.

Specificity

N. B.

From

x Rs an N.

Opacity does not imply intensionality

infer:

(= invalidity of *Extensionality*)

Some (specific) individual is Red by x.

2. Transparency and Specificity

***R* is <u>opaque</u>**

if the above inference schema is invalid.

2. Opacity and Monotonicity

***R* is <u>opaque</u>**

if the above inference schema is invalid.

Examples

(0a) I owe you a horse. Buridanus (1350), Geach (1965)

(b) Jones is seeking a lion. Quine (1956)

(c) Tom’s horse resembles a unicorn. Zimmermann (1993)

(d) Jones hired an assistent. Moltmann (1997)

2. Opacity and Monotonicity

***R* is <u>opaque</u>**

if the above inference schema is invalid.

Examples

(0a) I **owe** you a horse.

(b) Jones **is seeking** a lion.

(c) Tom's horse **resembles** a unicorn.

(d) Jones **hired** an assistent.

(0a) I owe you a horse.
(b) Jones is seeking a lion.
(c) Tom's horse resembles unicorn.
(d) Jones hired an assistent.

Paraphrases in terms of strange objects
(0a') I owe you an unspecific horse.
(b') Jones is seeking an intentional lion.
(c') Tom's horse resembles a generic unicorn.
(d') Jones hired a would-be assistent.

Paraphrases in terms of propositinal attitudes
(0a") I am obliged to see to it that it will be the case that I give you a horse.
(b") Jones is trying for it to be the case that Jones finds a lion.
(c") Given its outward appearance, Tom's horse could be a unicorn.
(d") Jones saw to it that there was someone who is an assistant.

(1) Jones is looking for a green sweater.
∴ Jones is looking for a sweater.

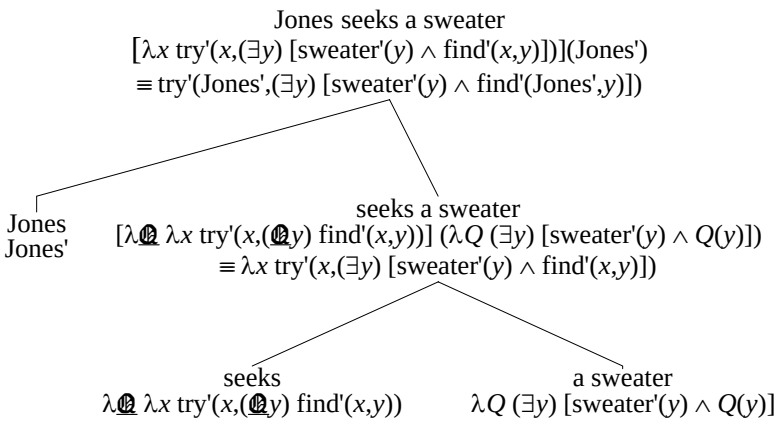
(2) Jones is wearing a green sweater.
∴ Jones is wearing a sweater.

(M↑) *x* is looking for a *P*. upward monotonicity
∴ *x* is looking for a *Q*. [$P \sqsubseteq Q$]

(M↓) *x* is looking for a *Q*.
∴ *x* is looking for a *P*. [$P \sqsubseteq Q$]

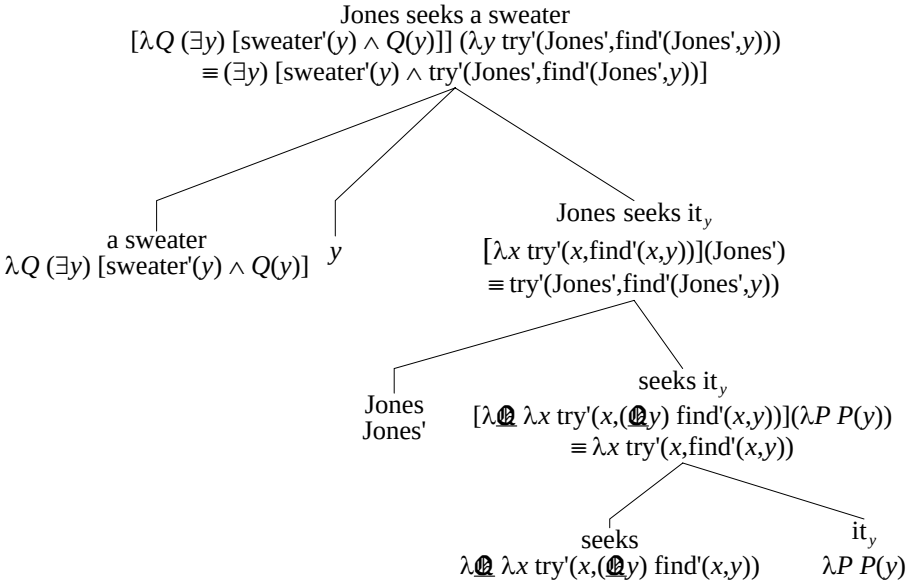
(3) $\text{seek}'(x,\textcircled{2}) = \text{try}'(x,(\textcircled{2}y)\ \text{find}'(x,y))$ Quine (1958, 1960), Montague (1968, 1970)

(4)



(5) Every man loves a woman.

(6)


Jones seeks a sweater
[λQ (∃y) [sweater'(y) ∧ Q(y)]] (λy try'(Jones',find'(Jones',y)))
≡ (∃y) [sweater'(y) ∧ try'(Jones',find'(Jones',y))]
a sweater
λQ (∃y) [sweater'(y) ∧ Q(y)]
y
Jones seeks it_y
[λx try'(x,find'(x,y))](Jones')
≡ try'(Jones',find'(Jones',y))
Jones
Jones'
seeks it_y
[λ□ λx try'(x,(□y) find'(x,y))](λP P(y))
≡ λx try'(x,find'(x,y))
seeks
λ□ λx try'(x,(□y) find'(x,y))
it_y
λP P(y)

(8) *Quine + Hintikka Analysis* cf. Hinitikka (1969)
seek'(x,𝒬)
[iff try'(x,(𝒬y) find'(x,y))]
iff being found by x is in the extension of 𝒬 whenever x's search is successful.

(9) *Success-Oriented Analysis* Moltmann (1997)
seek'(x,𝒬)
iff x's search is successful whenever being found by x is in the extension of 𝒬.

(10) *Exact Match Analysis*
seek'(x,𝒬)
iff x's search is successful just in case being found by x is in the extension of 𝒬.

(12) Jones is looking for a green sweater.
∴ Jones is looking for something.

(13) something … = (∃*x*) …

(12) Jones is looking for a green sweater.
∴ Jones is looking for something.

(13) something … = (∃*x*) …

(14b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ try'(Jones',(∃*y*) find'(Jones,*y*)) unspecific

(12) Jones is looking for a green sweater.
∴ Jones is looking for something.

(13) something … = (∃*x*) …

(14b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ try'(Jones',(∃*y*) find'(Jones,*y*)) unspecific

(15b) (∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ try'(Jones',find'(Jones',*y*))]
∴ (∃*y*) try'(Jones',find'(Jones,*y*)) specific

(12) Jones is looking for a green sweater.
∴ Jones is looking for something.

(13) something … = (∃*x*) …

(14b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ try'(Jones',(∃*y*) find'(Jones,*y*)) unspecific

(15b) (∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ try'(Jones',find'(Jones',*y*))]
∴ (∃*y*) try'(Jones',find'(Jones,*y*)) specific

(16b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ (∃□) try'(Jones',(□*y*) find'(Jones',*y*)) underspecific

(12) Jones is looking for a green sweater.
∴ Jones is looking for something.

(13) something … = (∃*x*) …

(14b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ try'(Jones',(∃*y*) find'(Jones,*y*)) unspecific

(15b) (∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ try'(Jones',find'(Jones',*y*))]
∴ (∃*y*) try'(Jones',find'(Jones,*y*)) specific

(16b) try'(Jones',(∃*y*)[sweater'(*y*) ∧ green'(*y*) ∧ find'(Jones',*y*)])
∴ (∃[illegible]) try'(Jones',([illegible]*y*) find'(Jones',*y*)) underspecific

(17) something … = (∃[illegible]) …

(18) Jones is looking for something Smith is looking for.

(21b) $(\exists y)$ [try'(Smith',find'(Smith',y)) $\wedge$ try'(Jones',find'(Jones',y))]

(22b) try'(Jones',$(\exists y)$[try'(Smith',find'(Smith',y)) $\wedge$ find'(Jones',y)])

(23b) $(\exists\mathfrak{Q})\ [\text{try}'(\text{Smith}',(\mathfrak{Q}y)\ \text{find}'(\text{Smith}',y)) \wedge \text{try}'(\text{Jones}',(\mathfrak{Q}y)\ \text{find}'(\text{Jones}',y))]$

(23b) $(\exists \mathfrak{A})$ [try'(Smith',$(\mathfrak{A}y)$ find'(Smith',y)) $\wedge$ try'(Jones',$(\mathfrak{A}y)$ find'(Jones',y))]

(24) I owe you nothing.

$\therefore$ I owe you something.

(27b) $(\exists \mathfrak{A})$[try'(Smith',$(\mathfrak{A}y)$ find'(Smith',y)) $\wedge$ $(\mathfrak{A}y)$ try'(Jones',find'(Jones',y))]

(∃) x is looking for a P.
∴ x is looking for something.

(∃b) try'(x,($\exists y$)[$P(y) \wedge$ find'(x,y)])
∴ ($\exists$𝔘) try'(x,(𝔘y) find'(x,y))

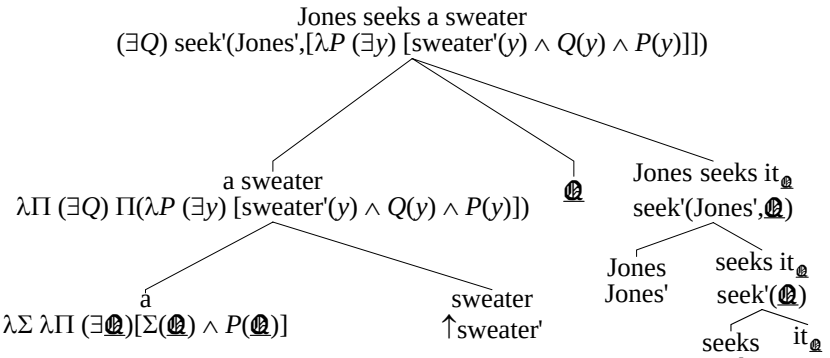
(28) Jones is looking for a green sweater.
Smith is looking for a pink hat.
∴ Jones is looking for something Smith is looking for (too).

(CO) *x* is looking for a *P*. inference to a common objective
y is looking for a *Q*.
∴ *x* is looking for something *y* is looking for.

(32) $\underline{\text{seek'(Jones',}[\lambda P\ (\exists y)[\text{sweater'}(y) \wedge \text{green'}(y) \wedge P(y)]])}$
$\therefore$ $(\exists \textcircled{2})[\text{sweater'}(\textcircled{2}) \wedge \text{seek'(Jones',}\textcircled{2})]$

(33) $\underline{\text{seek'(Jones',}[\lambda P\ (\exists y)[\text{sweater'}(y) \wedge \text{green'}(y) \wedge P(y)]])}$
$\therefore$ $(\exists \textcircled{2})[\uparrow\text{sweater'}(\textcircled{2}) \wedge \text{seek'(Jones',}\textcircled{2})]$

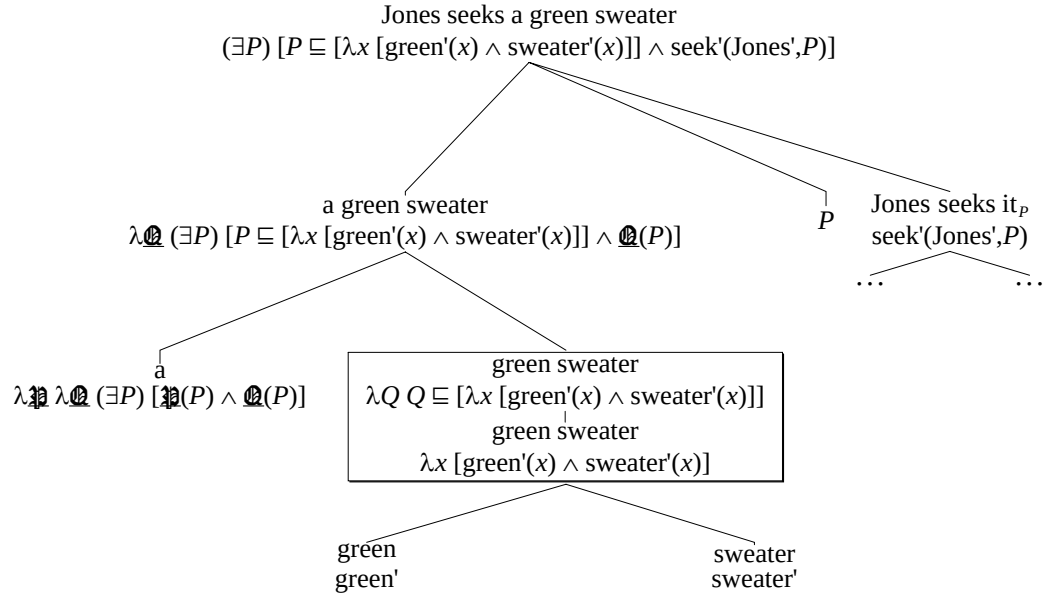
(39)



(40) Jones is looking for a green sweater, but Jones is not looking for a sweater.

(45) *Exact Match Analysis (type adaptation)*
seek'(x,P)
iff x's search is successful just in case being found by x is in the extension of $\exists_P$.

(50)



(51) (∃*P*) [*P* ⊑ sweater'(*x*) ∧ seek'(Jones',*P*)]

- *Unspecific objects*
  seek expresses a relation between a subject (the seeker) and an unspecific object of search.

- *Exact match*
  The relation expressed by *seek* holds true if the seeker's goal is reached just in case (s)he finds a specific object with the unspecific object as a property.

- *Type coercion*
  The indefinite object is re-interpreted as (existentially) quantifying over unspecific objects that are more general than the property expressed by its restrictor.

4.1 Specific Readings of Indefinites

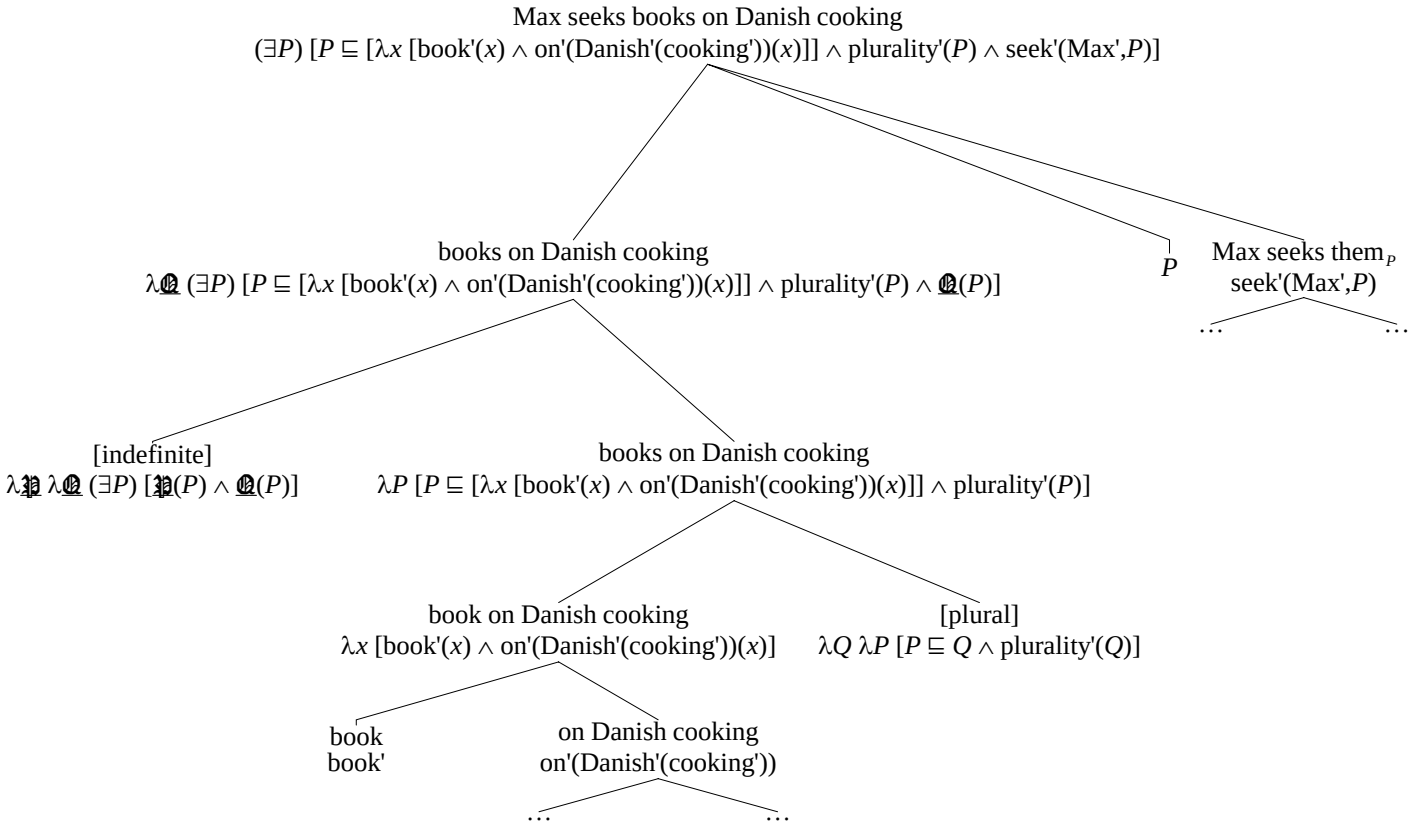
(66) *Argument Lowering [naive version]* Zimmermann (1993)
If $\Re$ is a relation between subjects and unspecific objects, then the *de re construal of* $\Re$ is that relation that holds between individuals x and y whenever $\Re$ holds between x and being y.

(66') *Argument Lowering [Kaplanian version]* Kaplan (1969, Zimmermann (to appear)
If $\Re$ is a relation between subjects and unspecific objects, then the *de re construal of* $\Re$ is that relation that holds between individuals x and y whenever $\Re$ holds between x and an unspecific object that is vivid for x and individuates y.

($\exists$) x is looking for a P.
$\therefore$ There is at least one P.

(81a) Max is looking for a book on Danish cooking.
(b) Max is looking for books on Danish cooking.

(82)

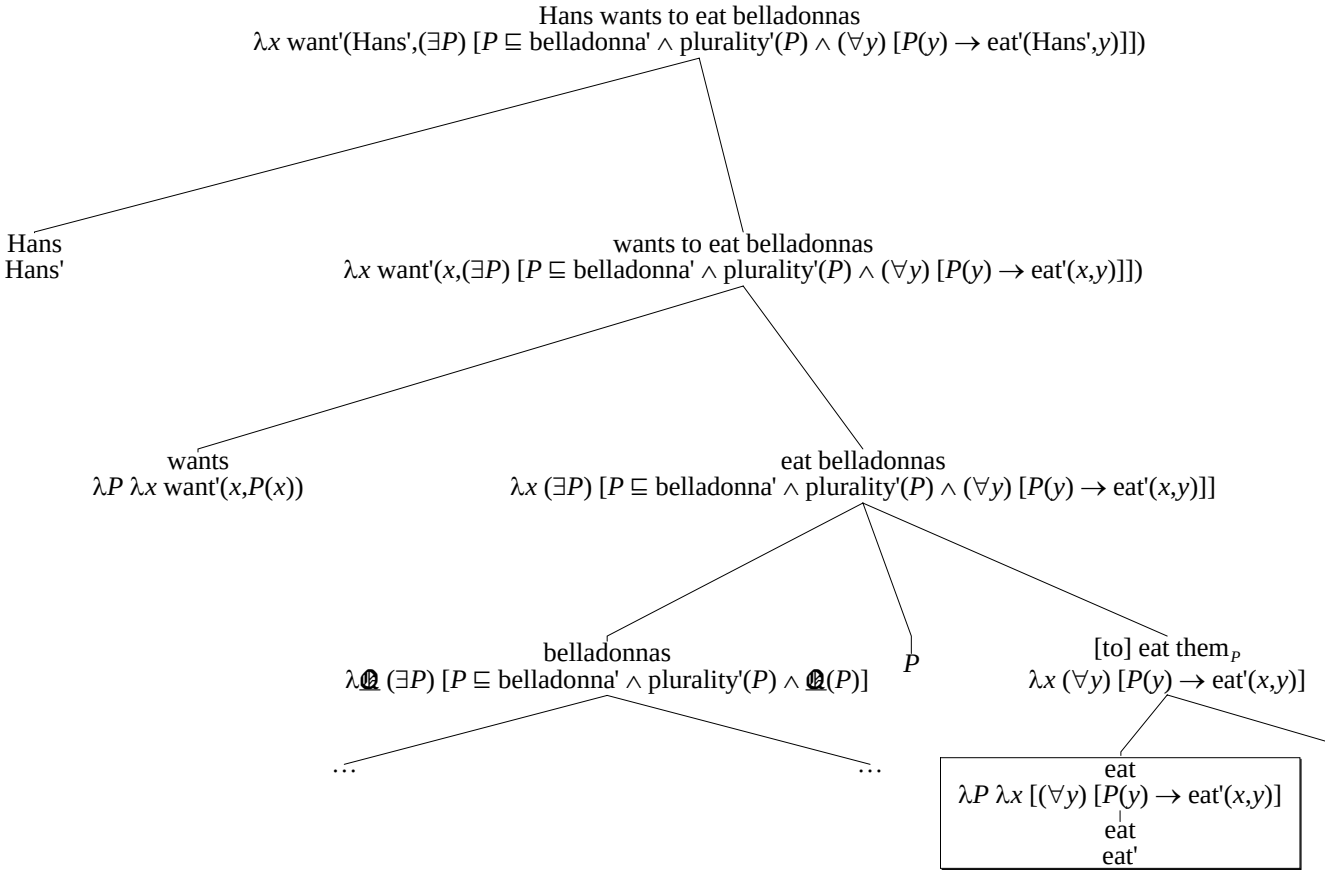


(83) plurality' = $\lambda P \, \square \, (\exists x) \, (\exists x) \, [P(x) \wedge P(y) \wedge x \neq y]$

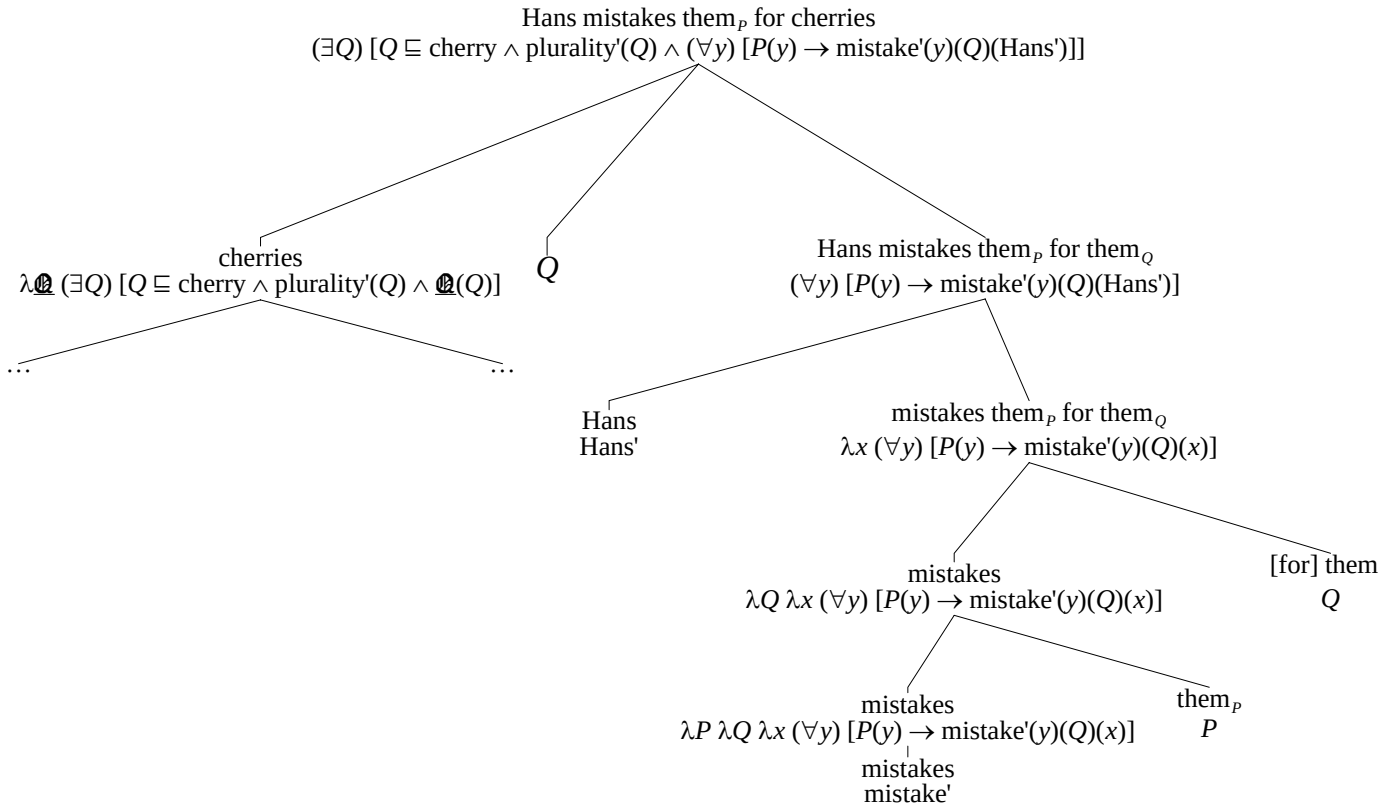
(84) Hans wanted to put belladonnas into the fruit salad, because he mistook them for [real] cherries.

(85) Hans wants to eat belladonnas and Hans mistakes them for cherries.

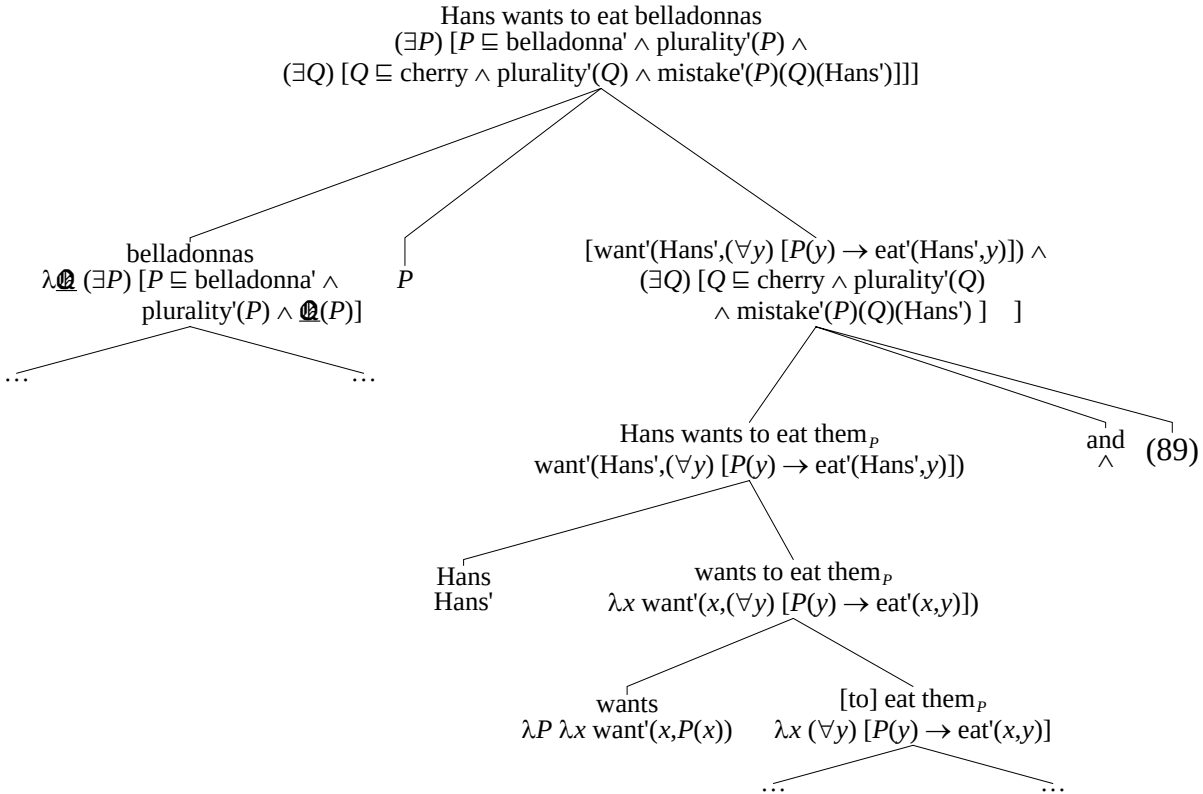
(87)


Hans wants to eat belladonnas
λx want'(Hans',(∃P) [P ⊑ belladonna' ∧ plurality'(P) ∧ (∀y) [P(y) → eat'(Hans',y)]])
Hans
Hans'
wants to eat belladonnas
λx want'(x,(∃P) [P ⊑ belladonna' ∧ plurality'(P) ∧ (∀y) [P(y) → eat'(x,y)]])
wants
λP λx want'(x,P(x))
eat belladonnas
λx (∃P) [P ⊑ belladonna' ∧ plurality'(P) ∧ (∀y) [P(y) → eat'(x,y)]]
belladonnas
λ (∃P) [P ⊑ belladonna' ∧ plurality'(P) ∧ (P)]
P
[to] eat them$_P$
λx (∀y) [P(y) → eat'(x,y)]
…
…
eat
λP λx [(∀y) [P(y) → eat'(x,y)]
eat
eat'

(89)


Hans mistakes them$_P$ for cherries
$(\exists Q)\,[Q \sqsubseteq \text{cherry} \wedge \text{plurality}'(Q) \wedge (\forall y)\,[P(y) \rightarrow \text{mistake}'(y)(Q)(\text{Hans}')]]$
cherries
$\lambda \square\,(\exists Q)\,[Q \sqsubseteq \text{cherry} \wedge \text{plurality}'(Q) \wedge \square(Q)]$
…
…
Q
Hans mistakes them$_P$ for them$_Q$
$(\forall y)\,[P(y) \rightarrow \text{mistake}'(y)(Q)(\text{Hans}')]$
Hans
Hans'
mistakes them$_P$ for them$_Q$
$\lambda x\,(\forall y)\,[P(y) \rightarrow \text{mistake}'(y)(Q)(x)]$
mistakes
$\lambda Q\,\lambda x\,(\forall y)\,[P(y) \rightarrow \text{mistake}'(y)(Q)(x)]$
[for] them
Q
mistakes
$\lambda P\,\lambda Q\,\lambda x\,(\forall y)\,[P(y) \rightarrow \text{mistake}'(y)(Q)(x)]$
them$_P$
P
mistakes
mistake'

(90) mistake' = $\lambda y\, \lambda P\, \lambda x$ [believe'(x,$P(y)$) $\wedge$ $\neg P(y)$]


Hans wants to eat belladonnas
(∃P) [P ⊑ belladonna' ∧ plurality'(P) ∧
(∃Q) [Q ⊑ cherry ∧ plurality'(Q) ∧ mistake'(P)(Q)(Hans')]]]
belladonnas
λ□ (∃P) [P ⊑ belladonna' ∧
plurality'(P) ∧ □(P)]
P
[want'(Hans',(∀y) [P(y) → eat'(Hans',y)]) ∧
(∃Q) [Q ⊑ cherry ∧ plurality'(Q)
∧ mistake'(P)(Q)(Hans')]]
Hans wants to eat them_P
want'(Hans',(∀y) [P(y) → eat'(Hans',y)])
and
∧
(89)
Hans
Hans'
wants to eat them_P
λx want'(x,(∀y) [P(y) → eat'(x,y)])
wants
λP λx want'(x,P(x))
[to] eat them_P
λx (∀y) [P(y) → eat'(x,y)]