

# Scope-taking and presupposition satisfaction

University of Chicago

June 13, 2019

# Presupposition

## The empirical observation

(1) Karlos **stopped** smoking.

$\leadsto$  Karlos smoked.

(presupposition)

We have linguistic devices that grammatically encode what we take for granted in making an utterance.

# Presupposition

## The empirical observation

How do we identify presuppositions?

- ▶ family-of-sentence tests (Chierchia and McConnell-Ginet, 1990)
- ▶ “hey, wait a minute!” test (von Stechow, 2004)

**Major research question:** what grammatical properties of an expression give rise to its presuppositions?

A compositional account answers two questions:

- ▶ how do we grammatically encode presuppositions in simple expressions (presupposition triggers)?
- ▶ how do presuppositions project in complex expressions?
  - ▶ the “projection problem” (Langendoen and Savin, 1971)

## Characterizing and explaining presupposition projection

Much work has gone into simply characterizing presupposition projection behavior (Karttunen, 1973, 1974, i.a.).

Principled explanations have generally taken two approaches:

- ▶ a “pragmatic” approach
  - ▶ general pragmatic principles are sufficient to properly characterize the presuppositions of complex sentences in terms of the presuppositions of their parts
  - ▶ Gazdar 1979, Schlenker 2009, 2011, Lassiter 2012
- ▶ a “semantic” approach
  - ▶ presupposition projection is a matter of grammar
    - ▶ semantic composition

# Characterizing and explaining presupposition projection

## Semantic accounts

General features of semantic accounts:

- ▶ regard presupposition projection as an effect of meaning composition
- ▶ deliver **semantic presuppositions**
- ▶ depend on pragmatic principles to relate these to actual **observed presuppositions**

# Characterizing and explaining presupposition projection

## Semantic accounts

Today, I will defend a semantic account of presupposition projection.

It will share properties with other semantic accounts.

- ▶ quantifier scope accounts (Russell, 1905)
- ▶ multi-valued logic accounts (Beaver, 1999, 2001; Beaver and Krahmer, 2001, i.a.)
- ▶ “two-dimensional” accounts (Karttunen and Peters, 1979)
- ▶ presupposition as semantic definedness (Heim, 1983)
- ▶ presupposition as anaphoric potential
  - ▶ DRT-based: van der Sandt 1992
  - ▶ dependent type semantics: Bekki 2014, i.a.

# The satisfaction account of presupposition projection

The following account (like Heim 1983) relates an expression's presuppositions to its semantic definedness conditions.

- ▶ it is **dynamic**: sentences denote **context-change potentials**
  - ▶  $\llbracket S \rrbracket_{\mathcal{M}} = \phi$
  - ▶  $\llbracket D ; S \rrbracket_{\mathcal{M}} = \llbracket D \rrbracket_{\mathcal{M}} + \phi$
- ▶ the meaning of a sentence is defined (in some linguistic context) just in case its presuppositions are **satisfied** (in that context)
- ▶ an expression's presuppositions are just statements of these satisfaction conditions
- ▶ it is thus a “satisfaction” account (Karttunen, 1974; Stalnaker, 1974; Heim, 1983; Beaver, 2001; von Stechow, 2008, i.a.)

The rest of today:

- ▶ discuss a general problem for extant satisfaction accounts
  - ▶ “trapped presupposition triggers”
  - ▶ known in many guises as the “proviso problem” (Geurts, 1996)
- ▶ present my account
  - ▶ a satisfaction account
  - ▶ a three-valued logic account
  - ▶ a two-dimensional account
  - ▶ a scope-based account





# The proviso problem

## Conditional sentences

Geurts (1996) notes that certain sentences are predicted to have weaker presuppositions than they are, in fact, observed to have.

(2) If Theo is a scuba diver, he will bring his wetsuit.

$\leadsto$  If Theo is a scuba diver, he has a wetsuit.

(presupposition)

(3) If Theo has a sister, he will bring his wetsuit.

$\leadsto$  Theo has a wetsuit.

(presupposition)

# The proviso problem

## Conditional sentences

Accounts like Heim's predict the correct presupposition for (2), but too weak a presupposition for (3).

- If Theo has a sister, he has a wetsuit.

$$c + \llbracket \textit{if Theo has a sister he will bring his wetsuit} \rrbracket$$

$$= c \backslash ((c + \llbracket \textit{Theo has a sister} \rrbracket) \backslash (c + \llbracket \textit{Theo has a sister} \rrbracket + \llbracket \textit{he will bring his wetsuit} \rrbracket))$$

Defined in exactly those contexts in which

*if Theo has a sister he has a wetsuit*

is true.

# The proviso problem

## Conditional sentences

The presuppositions of a complex sentence are defined in terms of the **local contexts** of its presupposition triggers.

- ▶ in (2) and (3):  $c + \text{conditional antecedent}$

**Generalization:** the semantic presuppositions of a sentence are often **weaker** than its observed presuppositions.

# The proviso problem

## Strengthening and selection

Much work since Geurts 1996 has been geared towards solving this problem within satisfaction accounts.

Complete solutions must address two problems (Singh, 2007):

- ▶ a **strengthening problem**
  - ▶ what are the available ways of strengthening a semantic presupposition
- ▶ a **selection problem**
  - ▶ given the available ways of strengthening a semantic presupposition, which one is selected as the observed presupposition?

## Trapped presupposition triggers

The proviso problem is but a collection of instances of a larger generalization.

### ► **trapped presupposition triggers**

(4) A: My parents think John is in bed.

B: They think I am in bed, **too**.

↪ My parents think John is in bed. (presupposition)

(5) A: John is in bed.

B: My parents think I am in bed, **too**.

↪ John is in bed. (presupposition)

No relation of strength between the presuppositions.

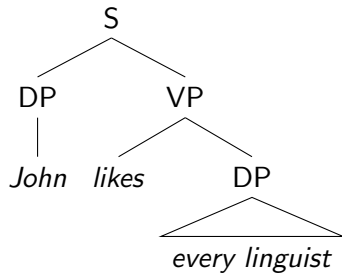
My solution:

Rather than regarding the “strengthening” problem as about strength, let us instead regard it as about **scope ambiguity**.

# Presupposition triggers and scope

## The basic idea

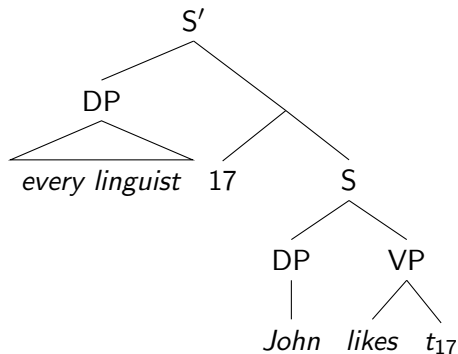
(6) John likes every linguist.



# Presupposition triggers and scope

## The basic idea

(6) John likes every linguist.



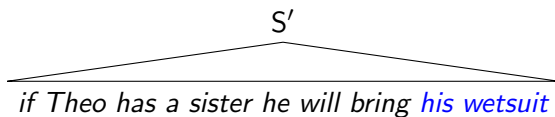
Quantifier Raising (Heim and Kratzer, 1998)



# Presupposition triggers and scope

## The basic idea

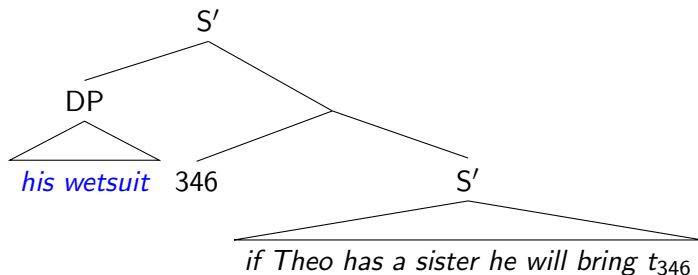
- (3) If Theo has a sister, he will bring *his wetsuit*.



# Presupposition triggers and scope

## The basic idea

- (3) If Theo has a sister, he will bring **his wetsuit**.



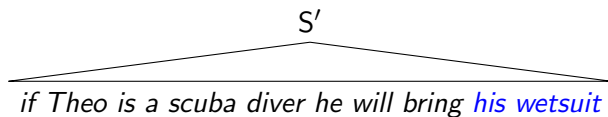
"Presupposition Trigger Raising"

↪ unconditional presupposition

# Presupposition triggers and scope

## The basic idea

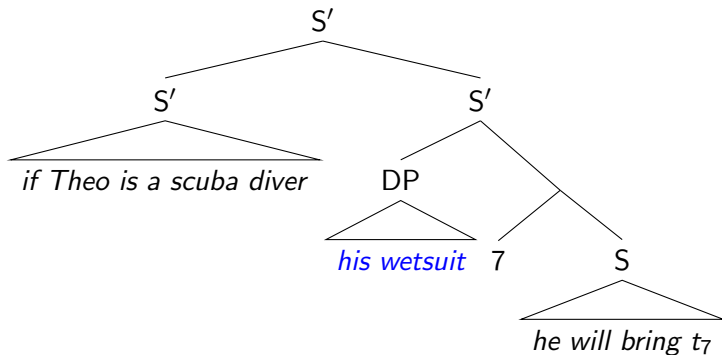
- (2) If Theo is a scuba diver, he will bring his wetsuit.



# Presupposition triggers and scope

## The basic idea

- (2) If Theo is a scuba diver, he will bring *his wetsuit*.

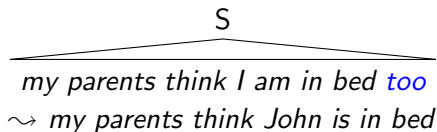
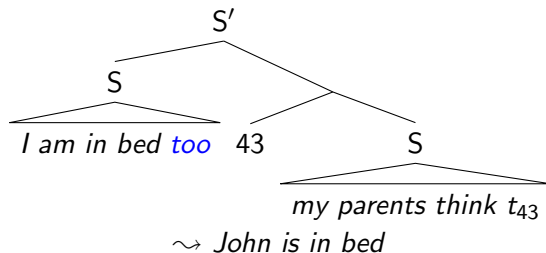


$\leadsto$  conditional presupposition

# Presupposition triggers and scope

## The basic idea

(4/5) My parents think I am in bed, *too*.



# Presupposition triggers and scope

## Outlook

How do we allow presupposition triggers to take scope?

- ▶ via the operators of a **graded monad** (Katsumata, 2014; Orchard et al., 2014; Fujii et al., 2016)

Why? It allows an account that is:

- ▶ compositional
- ▶ simply typed
- ▶ modular

Plan:

- ▶ introduce a basic, **presupposition-free** grammar
- ▶ add presuppositions via graded monadic **side effects**
  - ▶ conditionals
  - ▶ indefinites
- ▶ conclude

We model truth in terms of information states.

- ▶ functions from input assignments to sets of output assignments
- ▶ meanings will normally be **tests** (Groenendijk and Stokhof, 1991)
  - ▶ **true** sentences send input assignments out unmodified
  - ▶ **false** sentences throw them out and return the empty set
- ▶  $\mathcal{T} ::= e \mid v \mid a \mid t \mid \mathcal{T} \rightarrow \mathcal{T}$
- ▶  $\text{true} := (\lambda g. \{g\}): T := a \rightarrow a \rightarrow t$
- ▶  $\text{false} := (\lambda g. \{\}): T$

(7) Mary slept.

$$\llbracket (7) \rrbracket_{\mathcal{M}} = (\lambda g. \{g \mid \mathbf{sleepm}\})$$

$$\llbracket \text{Mary} \rrbracket_{\mathcal{M}} = (\lambda g, g'. \mathbf{m}): E := a \rightarrow a \rightarrow e$$

$$\llbracket \text{slept} \rrbracket_{\mathcal{M}} = (\lambda x, g. \{g \mid \mathbf{sleep}(xgg)\}): E \rightarrow T$$

$$\begin{aligned}\llbracket \text{Mary slept} \rrbracket_{\mathcal{M}} &= \llbracket \text{Mary} \rrbracket_{\mathcal{M}} \triangleleft \llbracket \text{slept} \rrbracket_{\mathcal{M}} \quad ((\triangleleft): \text{backward app.}) \\ &= (\lambda g, g'. \mathbf{m}) \triangleleft (\lambda x, g. \{g \mid \mathbf{sleep}(xgg)\}) \\ &= (\lambda x, g. \{g \mid \mathbf{sleep}(xgg)\})(\lambda g, g'. \mathbf{m}) \\ &= (\lambda g. \{g \mid \mathbf{sleep}((\lambda g, g'. \mathbf{m})gg)\}) \\ &= (\lambda g. \{g \mid \mathbf{sleepm}\})\end{aligned}$$



(1) Karlos **stopped** smoking.

*Karlos* and *smoking*:

$$\llbracket \textit{Karlos} \rrbracket_{\mathcal{M}} = (\lambda g, g'. \mathbf{k}): E$$

$$\llbracket \textit{smoking} \rrbracket_{\mathcal{M}} = (\lambda e, g. \{g \mid \mathbf{smoke}(egg)\}): V \rightarrow T$$
$$(V := a \rightarrow a \rightarrow v)$$

*stopped*?

- ▶ **anaphoric** to an event
- ▶ **presupposition**: the event satisfies the property denoted by its complement (*smoking*)
- ▶ **at-issue meaning**: the referent of its subject ended the event to which it is anaphoric

# Dynamic presupposition satisfaction

## Modelling definedness conditions

Presupposition success/failure:

- ▶ a new atomic type  $\#$ , with one inhabitant (also  $\#$ )
- ▶ meanings which may be either successes or failures:

- ▶ a successful individual (Karlos):

$$(\lambda s, f. s(\lambda g, g'. \mathbf{k})) : (E \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$$

- ▶ a failed individual:

$$(\lambda s, f. f \#) : (E \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$$

An  $\alpha$  with presuppositions:  $(\alpha \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$ .

# Dynamic presupposition satisfaction

## Modelling definedness conditions

Representing an  $\alpha$  with presuppositions:

- ▶  $(\top \Vdash M) = (\lambda s, f. sM): (\alpha \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$
- ▶  $(\perp \Vdash M) = (\lambda s, f. f\#): (\alpha \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$

$(\phi \Vdash M)$

- ▶ returns  $M$  when  $\phi$  is true
- ▶ returns  $\#$  when  $\phi$  is false

What about anaphora?

We introduce a function  $\mathbf{P}: \mathcal{T}^* \rightarrow \mathcal{T} \rightarrow \mathcal{T}$ .

$\mathcal{T}^*$ : the set of strings over  $\mathcal{T}$

- ▶ possible string of types of anaphoric dependencies

$$\begin{aligned}\mathbf{P}e\alpha &= (\alpha \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o \\ \mathbf{P}(ae)\alpha &= a \rightarrow \mathbf{P}e\alpha\end{aligned}$$

$$\llbracket \textit{stopped} \rrbracket_{\mathcal{M}}: (V \rightarrow T) \rightarrow \mathbf{P}V(E \rightarrow T)$$

$$= (\lambda P, e. (Pe = \text{true} \Vdash (\lambda x, g. \{g \mid (\exists e'. \text{end}(egg)(xgg)e')\})))$$

What about anaphora?

We introduce a function  $\mathbf{P}: \mathcal{T}^* \rightarrow \mathcal{T} \rightarrow \mathcal{T}$ .

$\mathcal{T}^*$ : the set of strings over  $\mathcal{T}$

- ▶ possible string of types of anaphoric dependencies

$$\mathbf{P}e\alpha = (\alpha \rightarrow o) \rightarrow (\# \rightarrow o) \rightarrow o$$

$$\mathbf{P}(ae)\alpha = a \rightarrow \mathbf{P}e\alpha$$

$$\llbracket \text{stopped} \rrbracket_{\mathcal{M}}: (V \rightarrow T) \rightarrow \mathbf{P}V(E \rightarrow T)$$

$$= (\lambda P. (P\mathbf{e} = \text{true} \Vdash_e (\lambda x, g. \{g \mid (\exists e'. \text{end}(\mathbf{e}gg)(xgg)e')\})))$$

# Dynamic presupposition satisfaction

## Modelling presupposition satisfaction

Meanings of type  $\mathbf{P}e\alpha$ :

- ▶ definedness conditions
- ▶ anaphora

What about presupposition satisfaction?

- ▶ we need to allow presuppositions to depend on prior context
- ▶ the effect of a context is to **change** or **filter** an expression's presuppositions
- ▶ we model this effect explicitly by adding a dependency to presupposition triggers of type  $T \rightarrow T$

$$\mathbf{D}e\alpha = (T \rightarrow T) \rightarrow \mathbf{P}e\alpha$$

# Dynamic presupposition satisfaction

## Modelling presupposition satisfaction

*stopped* (old):

- ▶  $(\lambda P. (Pe = \text{true} \Vdash_e (\lambda x, g. \{g \mid (\exists e'. \mathbf{end}(egg)(xgg)e')\})))$   
 $(V \rightarrow T) \rightarrow \mathbf{P}V(E \rightarrow T)$

*stopped* (new):

- ▶  $(\lambda P, c.$   
 $(c(Pe) = \text{true} \Vdash_e (\lambda x, g. \{g \mid (\exists e'. \mathbf{end}(egg)(xgg)e')\})))$   
 $(V \rightarrow T) \rightarrow \mathbf{D}V(E \rightarrow T)$
- ▶ abbrev: stop

# Dynamic presupposition satisfaction

## Semantic composition

$$\begin{aligned}\llbracket \textit{Karlos} \rrbracket_{\mathcal{M}} &= \mathbf{k} \\ &:= (\lambda g, g'. \mathbf{k}): E \\ \llbracket \textit{smoking} \rrbracket_{\mathcal{M}} &= \mathbf{smk} \\ &:= (\lambda e, g. \{g \mid \mathbf{smoke}(egg)\}): V \rightarrow T \\ \llbracket \textit{stopped} \rrbracket_{\mathcal{M}} &= \mathbf{stop}: (V \rightarrow T) \rightarrow \mathbf{D}V(E \rightarrow T)\end{aligned}$$

How do we compose such meanings?

- ▶  $\llbracket \textit{stopped smoking} \rrbracket_{\mathcal{M}} = \mathbf{stop} \triangleright \mathbf{smk}: \mathbf{D}V(E \rightarrow T)$   
(( $\triangleright$ ): forward app.)
- ▶ then what?



# Dynamic presupposition satisfaction

## Graded monads

Something special about **D** (and **P**):

- ▶ it is a graded monad

There are three operators:  $(\cdot^{\uparrow}_{\mathbf{D}})$ ,  $(\cdot^{\downarrow}_{\mathbf{D},f})$ , and  $\mu_{\mathbf{D},e,f}$ , satisfying specific laws.  $(e, f \in \mathcal{T}^*)$

Defined as:

- ▶  $M^{\uparrow}_{\mathbf{D}} = (\lambda c. (\top \Vdash M))$
- ▶  $(\lambda c. (\phi \Vdash_{x_1, \dots, x_m} M))^{\downarrow}_{\alpha_1 \dots \alpha_m, \beta_1 \dots \beta_n} (\lambda c. (\psi \Vdash_{y_1, \dots, y_n} N))$   
 $= (\lambda c. (\phi \wedge \psi \Vdash_{x_1, \dots, x_m, y_1, \dots, y_n} MN))$
- ▶  $\mu_{\mathbf{D}, \alpha_1 \dots \alpha_m, \beta_1 \dots \beta_n} (\lambda c. (\phi \Vdash_{x_1, \dots, x_m} (\lambda c'. (\psi \Vdash_{y_1, \dots, y_n} M))))$   
 $= (\lambda c. (\phi \wedge \psi[c/c'] \Vdash_{x_1, \dots, x_m, y_1, \dots, y_n} M[c/c']))$

# Dynamic presupposition satisfaction

## Graded monads

$$M^{\uparrow}_{\mathbf{D}} = (\lambda c. (\top \Vdash M))$$

- ▶ “unit” (or “return”)
- ▶ injects a value into the graded monad  $\mathbf{D}$  by giving it trivial **side effects** (i.e., presuppositions)

# Dynamic presupposition satisfaction

## Graded monads

$$\begin{aligned} (\lambda c. (\phi \Vdash_{x_1, \dots, x_m} M)) \overset{\mathbf{D}}{\downarrow}_{\alpha_1 \dots \alpha_m, \beta_1 \dots \beta_n} (\lambda c. (\psi \Vdash_{y_1, \dots, y_n} N)) \\ = (\lambda c. (\phi \wedge \psi \Vdash_{x_1, \dots, x_m, y_1, \dots, y_n} MN)) \end{aligned}$$

- ▶ “sequential application”
- ▶ applies a function with presuppositions to an argument with presuppositions
- ▶ sequences their side effects (i.e., presuppositions)

# Dynamic presupposition satisfaction

## Graded monads

$$\begin{aligned}\mu\mathbf{D}_{\alpha_1 \dots \alpha_m, \beta_1 \dots \beta_n}(\lambda c.(\phi \Vdash_{x_1, \dots, x_m} (\lambda c'.(\psi \Vdash_{y_1, \dots, y_n} M)))) \\ = (\lambda c.(\phi \wedge \psi[c/c'] \Vdash_{x_1, \dots, x_m, y_1, \dots, y_n} M[c/c']))\end{aligned}$$

- ▶ “join”
- ▶ collapses two layers of side effects (i.e., presuppositions) into a single layer
- ▶ an expression with presuppositions with presuppositions becomes just an expression with presuppositions

Unit, sequential application, and join work together to free trapped presupposition triggers.

# Dynamic presupposition satisfaction

## Building sentence meanings

Functional application can be modelled as **graded monadic functional application**.

$$\triangleright (\triangleright^*) ::= (\triangleright) \mid (\lambda u, v. ((\triangleright^*)^{\uparrow \mathbf{D}}_{\epsilon, e} u)^{\downarrow \mathbf{D}}_{e, f} v)$$

$$\triangleright (\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v. ((\triangleleft^*)^{\uparrow \mathbf{D}}_{\epsilon, e} u)^{\downarrow \mathbf{D}}_{e, f} v)$$

Two “inference” rules:

$$\triangleright \frac{M}{M^{\uparrow \mathbf{D}}_{\epsilon}} \quad \frac{m}{(\lambda x. x^{\uparrow \mathbf{D}}_{\epsilon})^{\uparrow \mathbf{D}}_{\epsilon, e_1} m} \quad \frac{m}{(\lambda x. x^{\uparrow \mathbf{D}}_{\epsilon})^{\uparrow \mathbf{D}}_{\epsilon, e_1} (\lambda x. x^{\uparrow \mathbf{D}}_{\epsilon, e_2} m)} \quad \dots$$

$$\triangleright \frac{m}{\mu_{\mathbf{D}_{e, f}} m}$$

# Dynamic presupposition satisfaction

## Building sentence meanings

$$\begin{aligned}
 & \llbracket \textit{Karlos stopped smoking} \rrbracket_{\mathcal{M}} \\
 &= \llbracket \textit{Karlos} \rrbracket_{\mathcal{M}}^{\uparrow \mathbf{D}} \triangleleft^* (\llbracket \textit{stopped} \rrbracket_{\mathcal{M}} \triangleright \llbracket \textit{smoking} \rrbracket_{\mathcal{M}}) \\
 &= \mathbf{k}^{\mathbf{D}}^{\uparrow} \triangleleft^* (\text{stop} \triangleright \text{smk}) \\
 &= \mathbf{k}^{\mathbf{D}}^{\uparrow} \triangleleft^* (\lambda c. (c(\text{smke}) = \text{true} \Vdash_e (\lambda x, g. \{g \mid (\exists e'. \text{end}(egg)(xgg)e')\}))) \\
 &= ((\triangleleft)^{\uparrow \mathbf{D}}_{\downarrow \epsilon, \vee} \\
 &\quad (\lambda c. (c(\text{smke}) = \text{true} \Vdash_e (\lambda x, g. \{g \mid (\exists e'. \text{end}(egg)(xgg)e')\}))))^{\downarrow \mathbf{D}}_{\vee, \epsilon} \\
 &\quad \mathbf{k}^{\mathbf{D}}^{\uparrow} \\
 &= (\lambda c. (c(\text{smke}) = \text{true} \Vdash_e (\lambda g. \{g \mid (\exists e'. \text{end}(egg)\mathbf{k}e')\})))
 \end{aligned}$$

An expression with presuppositions of type  $\mathbf{D} \vee T$ .

(3) If Theo has a sister, he will bring his wetsuit.

Following, e.g., Barker and Shan (2008) and Charlow (2014), the semantics of conditionals is defined in terms of:

- ▶ discourse update: (;)
- ▶ dynamic negation: not

$$\text{if } mn := \text{not}(m ; \text{not } n)$$

# Dynamic presupposition satisfaction

## Conditionals

$$(\lambda c.(\phi \Vdash_{x_1, \dots, x_m} p)) ; (\lambda c.(\psi \Vdash_{y_1, \dots, y_n} q)) \\ := (\lambda c.(\phi \wedge \psi[(\lambda t.c(p \Rightarrow t))/c] \Vdash_{x_1, \dots, x_m, y_1, \dots, y_n} p + q))$$

- ▶  $(\Rightarrow): T \rightarrow T \rightarrow T$  is “dynamic implication”
  - ▶  $p \Rightarrow q := (\lambda g.\{g' \mid (\forall g''.g' \in pg) \rightarrow (\exists g''.g'' \in qg')\})$
- ▶  $(+)$  is information state update
  - ▶  $p + q := (\lambda g.\{g'' \mid (\exists g'.g' \in pg \wedge g'' \in qg')\})$

So we're just updating  $p$  with  $q$ , while weakening the presuppositions of  $q$  to depend on  $p$ .

Anaphoric dependencies are just combined.



# Dynamic presupposition satisfaction

## Conditionals

$$\begin{aligned} & \text{not}(\lambda c.(\phi \Vdash_{x_1, \dots, x_m} p)) \\ & := (\lambda c.(\phi \Vdash_{x_1, \dots, x_m} (\lambda g. \{g \mid pg = \emptyset\}))) \end{aligned}$$

- ▶  $\text{not}(\lambda c.(\phi \Vdash_{x_1, \dots, x_m} \text{true})) = (\phi \Vdash_{x_1, \dots, x_m} \text{false})$
- ▶  $\text{not}(\lambda c.(\phi \Vdash_{x_1, \dots, x_m} \text{false})) = (\phi \Vdash_{x_1, \dots, x_m} \text{true})$

# Dynamic presupposition satisfaction

## Conditionals

$$\begin{aligned}\llbracket \text{his} \rrbracket_{\mathcal{M}} &= \text{his} : (E \rightarrow T) \rightarrow \mathbf{D}EE \\ &:= (\lambda P, c. (c(\mathbf{have}(xgg)\mathbf{t} + Px) = \mathbf{true} \Vdash_x x))\end{aligned}$$

$\llbracket \text{if Theo has a sister he will bring his wetsuit} \rrbracket$

$$\begin{aligned}&= \mu_{\mathbf{D}E, \epsilon} ((\text{if} \triangleright \llbracket \text{Theo has a sister} \rrbracket_{\mathcal{M}}) \\ &\quad \triangleright^* ((\lambda x. x^{\uparrow \mathbf{D}})^{\uparrow \mathbf{D}}_{\mathbf{E}} \llbracket \text{he will bring his wetsuit} \rrbracket_{\mathcal{M}})) \\ &= (\lambda c. (c(\lambda g. \{g \mid \mathbf{have}(xgg)\mathbf{t} \wedge \mathbf{wetsuit}(xgg)\}) = \mathbf{true} \\ &\quad \Vdash (\lambda g. \{g \mid \mathbf{has\_sistert} \rightarrow \mathbf{bring}(xgg)\mathbf{t}\})))\end{aligned}$$

“inverse scope”

- ▶ the presupposition trigger *his wetsuit* has taken scope

# Dynamic presupposition satisfaction

## Conditionals

(2) If Theo is a scuba diver, he will bring his wetsuit.

$\llbracket \text{if Theo is a scuba diver he will bring his wetsuit} \rrbracket$

$= ((\text{if} \triangleright \llbracket \text{Theo is a scuba diver} \rrbracket_{\mathcal{M}}) \triangleright \llbracket \text{he will bring his wetsuit} \rrbracket_{\mathcal{M}})$

$= (\lambda c. (c(\lambda g. \{g \mid \mathbf{divert} \rightarrow (\mathbf{have}(xgg)\mathbf{t} \wedge \mathbf{wetsuit}(xgg)\})) = \mathbf{true}$   
 $\quad \Vdash (\lambda g. \{g \mid \mathbf{divert} \rightarrow \mathbf{bring}(xgg)\mathbf{t}\})))$

“surface scope”

May be smoothly integrated with a dynamic semantics for indefinites.

(8) A man walked in. The man sat down.

- ▶ second sentence presupposes a man
- ▶ presupposition is satisfied by the first sentence

$$\llbracket a \text{ man walked in} \rrbracket_{\mathcal{M}} = (\lambda g. \{g' \mid g[1]g' \wedge \mathbf{man}g'_1 \wedge \mathbf{walk}g'_1\})$$

$$\llbracket the \rrbracket_{\mathcal{M}} = (\lambda P, c. (c(P(\lambda g, g'.g_1)) = \mathbf{true} \Vdash (\lambda g, g'.g_1)))$$

$$\begin{aligned} \llbracket the \text{ man sat down} \rrbracket_{\mathcal{M}} \\ = (\lambda c. (c(\lambda g. \{g \mid \mathbf{man}g_1\}) = \mathbf{true} \Vdash (\lambda g. \{g \mid \mathbf{sit}g_1\}))) \end{aligned}$$

# Dynamic presupposition satisfaction

## Indefinites

$$\llbracket a \text{ man walked in. the man sat down.} \rrbracket_{\mathcal{M}}$$

$$= \llbracket a \text{ man walked in} \rrbracket_{\mathcal{M}}^{\uparrow} ; \llbracket the \text{ man sat down} \rrbracket_{\mathcal{M}}$$

$$= (\lambda c. (c((\lambda g. \{g' \mid g[1]g' \wedge \mathbf{man}g'_1 \wedge \mathbf{walk}g'_1\})))$$

$$\Rightarrow (\lambda g. \{g \mid \mathbf{man}g_1\})) = \mathbf{true}$$

$$\Vdash (\lambda g. \{g' \mid g[1]g' \wedge \mathbf{man}g_1 \wedge \mathbf{walk}g_1 \wedge \mathbf{sit}g_1\})))$$

$$= (\lambda c. (c\mathbf{true} = \mathbf{true} \Vdash (\lambda g. \{g' \mid g[1]g' \wedge \mathbf{man}g_1 \wedge \mathbf{walk}g_1 \wedge \mathbf{sit}g_1\}))))$$

“trivial” presuppositions

# Conclusions

Satisfaction accounts and “trapped presupposition triggers”:

- ▶ the “proviso problem” (more generally, that of “trapped presupposition triggers”) is **not** an inherent flaw of Heimian accounts of presupposition projection
- ▶ the problem disappears once the compositional repertoire is restructured in terms of the operators of a graded monad

Outlook:

- ▶ can the same basic technology be integrated with other dynamic semantic frameworks (e.g., de Groote, 2006; Charlow, 2014)?
- ▶ can it be modified to handle presupposition accommodation?
  - ▶ hybrid account with de Groote and Lebedeva (2010); Lebedeva (2012)?
- ▶ the **selection** problem as resolution of scope ambiguity

# References

- Barker, Chris, and Chung-chieh Shan. 2008. Donkey anaphora is in-scope binding. *Semantics and Pragmatics* 1:1–46.
- Beaver, David, and Emiel Krahmer. 2001. A partial account of presupposition projection. *Journal of Logic, Language and Information* 10:147–182.
- Beaver, David I. 1999. Presupposition accomodation: A plea for common sense. In *Logic, language, and computation*, ed. Lawrence S. Moss, Jonathan Ginzburg, and de Rijke Maarten, volume 2, 21–44. Stanford: CSLI Publications.
- Beaver, David I. 2001. *Presupposition and assertion in dynamic semantics*. Studies in Logic, Language and Information. Stanford: CSLI Publications.
- Bekki, Daisuke. 2014. Representing anaphora with dependent types. In *Logical aspects of computational linguistics*, ed. Nicholas Asher and Sergei Soloviev, volume 8535 of *Lecture Notes in Computer Science*, 14–29. Heidelberg: Springer.



# References

- Charlow, Simon. 2014. On the semantics of exceptional scope. Doctoral Dissertation, NYU, New York.
- Chierchia, Gennaro, and Sally McConnell-Ginet. 1990. *Meaning and grammar: An introduction to semantics*. Cambridge: MIT Press.
- von Fintel, Kai. 2004. Would you believe it? the king of France is back! presuppositions and truth-value intuitions. In *Descriptions and beyond*, ed. Marga Reimer and Anne Bezuidenhout, chapter 8, 269–296. Oxford: Oxford University Press.
- von Fintel, Kai. 2008. What is presupposition accommodation, again? *Philosophical Perspectives* 22.
- Fujii, Soichiro, Shin-ya Katsumata, and Paul-André Melliès. 2016. Towards a formal theory of graded monads. In *Proceedings of the 20th International Conference on Foundations of Software Science and Computation Structures*, ed. Bart Jacobs and Christof Löding, number 9634 in Lecture Notes in Computer Science, 513–530.

# References

- Gazdar, Gerald. 1979. *Pragmatics: Implicature, presupposition and logical form*. New York: Academic Press.
- Geurts, Bart. 1996. Local satisfaction guaranteed: A presupposition theory and its problems. *Linguistics and Philosophy* 19:259–294.
- Groenendijk, Jeroen A., and Martin B. J. Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy* 14:39–100.
- de Groote, Philippe. 2006. Towards a Montagovian account of dynamics. In *Proceedings of the 16th Conference on Semantics and Linguistic Theory*, ed. Masayuki Gibson and Jonathan Howell, 1–16. Ithaca, NY: Cornell.
- de Groote, Philippe, and Ekaterina Lebedeva. 2010. Presupposition accommodation as exception handling. In *Proceedings of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 71–74. Tokyo: Association for Computational Linguistics.

# References

- Heim, Irene. 1983. On the projection problem for presuppositions. In *Proceedings of the 2nd West Coast Conference on Formal Linguistics*, ed. Michael D. Barlow, Daniel P. Flickinger, and Nancy Wiegand, 114–125. Stanford: Stanford University Press.
- Heim, Irene, and Angelika Kratzer. 1998. *Semantics in generative grammar*. Malden: Blackwell.
- Karttunen, Lauri. 1973. Presuppositions of compound sentences. *Linguistic Inquiry* 4:169–193.
- Karttunen, Lauri. 1974. Presuppositions and linguistic context. *Theoretical Linguistics* 1:181–194.
- Karttunen, Lauri, and Stanley Peters. 1979. Conventional implicature. In *Syntax and semantics*, ed. Choon-Kyu Oh and David A. Dinneen, volume 11, 1–56. New York: Academic Press.

## References

- Katsumata, Shin-ya. 2014. Parametric effect monads and semantics of effect systems. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. San Diego.
- Langendoen, Donald Terence, and Harris B. Savin. 1971. The projection problem for presuppositions. In *Studies in linguistic semantics*, ed. Charles J. Fillmore and Donald Terence Langendoen, 55–60. Holt, Rinehart & Winston.
- Lassiter, Daniel. 2012. Presuppositions, provisos, and probability. *Semantics and Pragmatics* 5:1–37.
- Lebedeva, Ekaterina. 2012. Expressing discourse dynamics through continuations. Doctoral Dissertation, Université de Lorraine, Lorraine.
- Orchard, Dominic A., Tomas Petricek, and Alan Mycroft. 2014. The semantic marriage of monads and effects. *CoRR* abs/1401.5391. URL <http://arxiv.org/abs/1401.5391>.

# References

- Russell, Bertrand. 1905. On denoting. *Mind* 14:479–493.
- van der Sandt, Rob A. 1992. Presupposition projection as anaphora resolution. *Journal of Semantics* 9:133–177.
- Schlenker, Philippe. 2009. Local contexts. *Semantics and Pragmatics* 2:1–78.
- Schlenker, Phillippe. 2011. The proviso problem: a note. *Natural Language Semantics* 19:395–422.
- Singh, Raj. 2007. Formal alternatives as a solution to the proviso problem. In *Proceedings of the 17th Conference on Semantics and Linguistic Theory*, ed. Tova Friedman and Masayuki Gibson, 264–281. Ithaca, NY: Cornell.
- Stalnaker, Robert. 1974. Pragmatic presuppositions. In *Semantics and philosophy*, ed. Milton K. Munitz and Peter K. Unger, 197–214. New York: New York University Press.