# Presupposition Accommodation in a Constructive Update Semantics *

Joris Hulstijn
joris@cs.utwente.nl

**Abstract**

Presupposition concerns information that is assumed to be part of the discourse-context by the speaker, as apparent from the syntactic form of the utterance. The hearer, being cooperative, will normally fill in missing presupposed information, provided it is compatible with his/her version of the discourse-context. Several different approaches to this process of presupposition accommodation can be characterized using update semantics with an additional stack-structure. Semantic tableaux can be used to represent the information states of update semantics. Branches of tableaux are constructed during interpretation. Therefore the approach may be labeled a *constructive update semantics*.

## 1 Introduction

A sentence is not a random collection of words. Sentences are put together following syntactic rules. But not only syntax puts constraints on the use of words. Example (1) is wrong, because the verb 'to eat' expects the subject to be animate.[1]

(1) *The stone *ate* the cake.

Presuppositions are just such constraints on the use of a certain linguistic construction or choice of words. Constructions carrying such a constraint are called *presupposition triggers*. The presupposed information is part of the meaning of an utterance, just like asserted information. I assume a lexicon that lists both the asserted and presupposed information for all expressions.

There are several types of presupposition triggers. The verb 'to eat' is an example of a *lexical presupposition* trigger: the presupposition arises from the meaning of the word. Similar to lexical presuppositions are the

*Research for this paper was done with Henk Zeevat at $\alpha$-informatica, Faculty of Arts, University of Amsterdam. The paper has improved thanks to two anonymous referees.

[1] Wrong or infelicitous utterances are indicated by a *.

so called *factive verbs*, like ‘know’ and ‘regret’ that presuppose their complements to be true. On the other side of the scale we have the *existential presuppositions*, induced by *definite descriptions*, *proper names*, *quantifiers* and special *focusing constructions* like *clefts*. These referring expressions presuppose the existence of their referents. Words like ‘but’ or ‘even’ presuppose a more subtle sentence-structure. For a good overview of triggers see (Gazdar, 1979).

(2) a. John *regrets* that he killed his wife.
*pres:* John killed his wife.
*ass:* John feels remorse.

b. *Most* of Jack’s children are happy.
*pres:* Jack has children.
*ass:* Most of the children of Jack are happy.

c. *Even* Fred likes bananas.
*pres:* Fred is the least likely person to like something.
*ass:* Fred likes bananas.

What is the relevance of presuppositions to computational linguistics? One of the main problems in computational linguistics is the notion of *context*. Presupposition triggers reveal what the speaker takes to be part of the context. Exploiting this information, may help advance natural language understanding. The central notion of this paper is *presupposition accommodation*: the way in which the (hearer’s version of the) context is changed, when an utterance with a presupposition is added to the context (Lewis, 1979; Heim, 1983).

When seen as a form of inference, presupposition accommodation exhibits strange logical properties. This paper attempts to characterize these properties in a logical framework. I use *accommodation strategies* to model several well-known approaches to presupposition from the literature. In this way different proposals can be compared and evaluated in a single framework. (Section 3)

In Section 2 the logical framework is defined: *update semantics* (Veltman, 1996). Like DRT and other theories of dynamic semantics, update semantics conceives of meaning as the potential change it makes to the information in the context. But unlike DRT, update semantics views information as a way of distinguishing between different ways the world can be: *possible worlds*, *indices*, *situations* or whatever you might want to call them. Adding information means eliminating such epistemic possibilities. A set of epistemic possibilities represents information irrespective of the particular way in which it was formulated. By contrast, the representational nature of a DRS makes it difficult to deal with information that is implicit in the context. However, when we implement the framework in a computer program, this distinction becomes obsolete. Then an explicit representation needs to be constructed for changing information in a context. In Section 4 the implementation is discussed. Representations are constructed incrementally. When transitions between representations are modeled directly in the semantics, this might be called a *constructive update semantics*.

## 2 Update Semantics

Instead of modeling the context as the common ground between speaker and hearer, I simply model the *hearer's information state*. I assume that the hearer behaves as a *rational* agent. So information is to remain consistent at all time and stay closed under logical consequence.

In *Update Semantics* a formula is interpreted as a transition between information states. Formally an Update Semantics for a logical language $L$ is a *frame*$\{\langle \Sigma, [\phi]\}_{(\phi \in L)}\rangle\}$, consisting of a set of information states $\Sigma$ and a (partial) update function $[.]$ over $\Sigma$. Following Veltman (1996) I use a postfix notation: $\sigma' = \sigma[\phi]$ means that $\sigma'$ is the result of updating an information state $\sigma$ with a formula $\phi$. Updates are originally supposed to model the effect of *assertions*. In this paper special updates, indicated by the operator $\partial$,[2] will be defined to model *presuppositions*. I assume a grammar that translates utterances from (a fragment) of English into the logical language $L$. Interaction between presupposition and quantification results in interesting and complicated issues, e.g. (Heim, 1983). This paper leaves these issues aside. However, the analysis can be extended to deal with them (Hulstijn, 1995). So, the logical language is taken to be the normal language of predicate logic without quantification, extended with presupposition operator $\partial$[3]. The set of ground formulas is called the *vocabulary* $\mathcal{A}$.

DEFINITION 1 (logical language $L$)
*Given sets of $n$-ary predicates $Pred^n$ and constants $Const$, define*

$$\begin{array}{rcl} \mathcal{A} & := & Const \cup \{Pc_1 \ldots c_n | P \in Pred^n, c_i \in Const, (1 \leq i \leq n)\} \\ L & := & p|\neg\phi|\phi \wedge \psi|\phi \vee \psi|\bot|\top|\partial\phi \;\; (p \in \mathcal{A}, \phi, \psi \in L) \end{array}$$

Update semantics defines information states as *sets of epistemic possibilities*. Technically an epistemic possibility $w$ is a total valuation with respect to $\mathcal{A}$. So the set of all possibilities $W = \{w | w : \mathcal{A} \mapsto \{0,1\}\}$ and the set of all information states $\Sigma = pow(W)$. The semantics is defined inductively on all $\phi \in L$.

[2]Notation $\partial$ derived from Beaver (1993).
[3]Not every possible expression with $\partial$ makes sense. I assume the grammar will produce sensible formulas.

DEFINITION 2 (Update semantics)
*US is given by* $\langle \Sigma_{US}, [.] \rangle$, *where:*

$$\begin{array}{rcl} \sigma[p] & = & \{w \in \sigma | w(p) = 1\} \ (p \in \mathcal{A}) \\ \sigma[\neg\phi] & = & \sigma \setminus \sigma[\phi] \\ \sigma[\phi \wedge \psi] & = & \sigma[\phi][\psi] \\ \sigma[\partial\phi] & = & \textit{depends on accommodation strategy} \end{array}$$

In the atomic case, all worlds incompatible with asserted information $p$ are eliminated. Negation is modeled by set complement. Conjunction is modeled by function composition on updates. This gives a *sequential* notion of conjunction. Disjunction and implication can be defined using the standard equivalences $\phi \vee \psi \equiv \neg(\neg\phi \wedge \neg\psi)$ and $\phi \to \psi \equiv \neg(\psi \wedge \neg\psi)$. $\bot$ and $\top$ can be defined by $\bot \equiv p \wedge \neg p$ and $\top \equiv \neg\bot$ for arbitrary $p$. Since $Const \subseteq \mathcal{A}$ we can model existence of objects and existential presupposition.[4] For instance 'a man' is modeled by $c_3 \wedge man(c_3)$. 'John' is modeled by the formula $\partial(c_5 \wedge named(c_5, john))$ All possibilities $w$ for which $w(c_5) \neq 1$ will be eliminated.

The following additional notions are useful. The state of "no information yet", is called the *initial state* or $\mathbb{1}$. No information means that all possibilities are still open. So $\mathbb{1}$ is modeled by $W$.[5] An update with an inconsistent formula or with information contradicting present information results in the unique *absurd state* $\mathbb{0}$, modeled by $\emptyset$. Trivially $\mathbb{0}$ contains all information at once. There is a transitive and reflexive order $\sqsubseteq$ over information states, specifying *information growth.* So $\sigma \sqsubseteq \tau$ iff $\tau$ contains at least as much information as $\sigma$. Here $\sqsubseteq$ is modeled by $\supseteq$. For presuppositionless formulas, the update function preserves information growth.[6]

$$\forall \sigma, \phi : \mathbb{1} \sqsubseteq \sigma, \sigma \sqsubseteq \mathbb{0}, \sigma \sqsubseteq \sigma[\phi]$$

An information state $\sigma$ contains $\phi$ or *supports* $\phi$ when updating $\sigma$ with $\phi$ will not increase the information in $\sigma$.[7]

$$\sigma \Vdash \phi \text{ iff } \sigma[\phi] \sqsubseteq \sigma$$

Update Semantics has a non-classical view on validity and entailment. An argument is valid if, after having applied all premises in the right order to some information state, the conclusion is supported. The second notion of entailment, $\models^{\mathbb{1}}$, represents entailment with a *closed world assumption*: if

[4] Think of $Const$ as the set of nullary predicates. So $w(c) = 1$ iff $c$ exists at $w$.

[5] In applications $\mathbb{1}$, like all information states, will be constrained by meaning postulates representing domain knowledge.

[6] Proof by induction on $\phi$.

[7] By information growth we have $\sigma \sqsubseteq \sigma[\phi]$. When $\sqsubseteq$ is anti-symmetric, this definition becomes equivalent to $\sigma \Vdash \phi$ iff $\sigma[\phi] = \sigma$. All information orders based on $\subseteq$ are anti-symmetric.

the premises are *all* you learned, then the conclusion is supported. A combination of presupposition and $\models^{\mathbb{1}}$ may produce non-monotonic inference. (see Section 3.5)

$$\begin{array}{lcccrcc} \phi_1,\ldots,\phi_n & \models & \psi & \text{iff} & \forall\sigma\sigma[\phi_1]\ldots[\phi_n] & \Vdash & \psi \\ \phi_1,\ldots,\phi_n & \models^{\mathbb{1}} & \psi & \text{iff} & \mathbb{1}[\phi_1]\ldots[\phi_n] & \Vdash & \psi \end{array}$$

# 3 Presupposition Accommodation

In this section I will use four well-known empirical *observations* of the behavior of presupposition to guide the definitions of the accommodation strategies.

## 3.1 presupposition as precondition

Presuppositions can be seen as a *constraint* on their immediate context. A presupposition works very much like a *precondition* in mathematics or computer science. If the precondition fails, the meaning of the mathematical expression or the result of the program becomes undefined.

(3) $\sqrt{a}$
*pres:* $0 \leq a$

(4) a. *John's dog* is happy.
*pres:* John has a dog.

b. John is allergic to dogs.
**John's dog* is happy.

This notion of 'presupposition as precondition' is very similar to what I call the *classic* strategy: a sentence $S$ presupposes proposition $\phi$ in a context $\sigma$ iff $S$ can only be uttered felicitously provided $\sigma$ supports $\phi$. (Karttunen, 1974; Lewis, 1979; Stalnaker, 1979; Heim, 1983; Beaver, 1993)

The presupposition must be true in the context, otherwise the utterance is *infelicitous.* See for instance Example (4). In other words, unless the presupposition is satisfied, the conversational effect of the utterance is *undefined.* Strictly speaking the classic strategy is the strategy of *no* accommodation of the context.

STRATEGY 1 (classic)

$$\sigma[\partial\phi] \quad = \quad \begin{cases} \sigma & \mathit{if}\, \sigma \Vdash \phi \\ \mathit{undef} & \mathit{otherwise} \end{cases}$$

Since the update function models the conversational effect of an utterance, this means the update function has become a *partial function*: it's value is not defined for all $\phi$. I take it that function composition and information state subtraction, and therefore the logical operators $\wedge$ and $\neg$, are *strict* with respect to undefinedness.[8] Given update semantics, the classic *projection* behavior of (Karttunen, 1973) automatically comes out. In Section 3.6 this projection behavior will be changed. The projection behavior of disjunction is controversial.

$$\begin{array}{rcll} \sigma[\phi \wedge \psi] & = & undef & \text{if } \sigma[\phi] = undef \text{ or } (\sigma[\phi])[\psi] = undef \\ \sigma[\neg\phi] & = & undef & \text{if } \sigma[\phi] = undef \\ \sigma[\phi \rightarrow \psi] & = & undef & \text{if } \sigma[\phi] = undef \text{ or } (\sigma[\phi])[\psi] = undef \end{array}$$

## 3.2 accommodation proper

Preconditions are only part of the story. Discourse is a communicative action. Communication only succeeds on the assumption that all participants cooperate (Grice, 1975). Uttering a sentence of which the presupposition is known to be false is uncooperative; therefore the hearer will infer from a presupposition trigger that the speaker takes the presupposition to be true. Thus, the hearer will *adjust* his or her version of the context, by adding the presupposition. This adjustment is called *presupposition accommodation*(Lewis, 1979; Heim, 1983). Note however that accommodation proper is only allowed when the presupposition is compatible with the context. Otherwise, the utterance remains infelicitous and its effect undefined. Therefore I call this the *cautious accommodation* strategy: only add the presupposition when compatible.

STRATEGY 2 (cautious)

$$\sigma[\partial\phi] \quad = \quad \begin{cases} \sigma[\phi] & \textit{if } \sigma[\phi] \neq \mathbb{0} \\ undef & \textit{otherwise} \end{cases}$$

## 3.3 presupposition test

This adjustment of the context can be seen as a form of inference. Information is inferred from the trigger. It is, however, is a very strange kind of inference. The infelicity prediction/presuppositional inference normally remains when embedded under negation, modal operators or belief contexts. (Example 5)

(5) a. *John's dog* is happy.
*pres:* John has a dog.

[8] An operation $\bullet$ on information states $\sigma, \tau$ is *strict* if $(\sigma \bullet \tau) = undef$ iff $\sigma = undef$ or $\tau = undef$.

b. It is not the case that *John's dog* is happy.
*pres:* Johnhas a dog.

c. Maybe *John's dog* is happy.
*pres:* John has a dog.

(6) a. Bella is a cow.
*implies:* Bella mows.

b. It is not the case that Bella is a cow.
*implies:* ... ?

c. Maybe Bella is a cow.
*implies:* ... ? *possibly* Bella mows.

But when we negate the antecedent of a conditional, as in (6a), the conclusion is lost. When the antecedent is weakened by *maybe*, the conclusion is weakened too. None of this happens with presupposition. Therefore this observation can be used as a *presupposition test.* Note that the classic strategy already preserves the infelicity prediction under negation. But presuppositional inference is not preserved under both the classic and cautious strategies. This will be remedied in Section 3.6.

## 3.4 cancelling

On the other hand presuppositional inference is *defeasible.* The infelicity prediction/presuppositional inference will, in some particular embedding contexts, be *cancelled* or *bound.* This may happen for instance in indirect speech, when the antecedent of a conditional or the first of two conjuncts implies the presupposition, in belief contexts, or in some special negated sentences.[9] Here are some examples:

(7) John is allergic to dogs.

a. *If John buys a bone, *John's dog* is happy.

b. If John has a dog, *John's dog* is happy.

c. (But) John has a dog and *John's dog* is happy.

d. Mary believes that *John's dog* is happy.

[9]Traditionally such cancelling contexts are called *plugs*, as opposed to the *holes* that let presuppositions seep through (Karttunen, 1973). But, unlike Karttunen, I am able to deal with the context-dependency of most plugs. As shown by Gazdar (1979) there or no strict plugs.

In Example (7) the context suggested by John's allergy is incompatible with the presupposition. So as expected, (7b) is infelicitous. But in (7c) it is not. The infelicity prediction/presuppositional inference is said to be cancelled or bound. The difference is that in (7c) the antecedent already implies the presupposition. What seems to be the case here, is that interpretation shifts to a temporary context, corresponding to the antecedent. The temporary context supports the presupposition. There is no conflict with the overall context. Conjunction works in a similar way (Example (7d)). This effect of binding to the first conjunct or antecedent, is already covered by the classic or cautious strategy and the update semantics framework.

Belief contexts, as in Example (7e), also seem to create such a temporary context. Since Mary doesn't know of John's allergy, the embedded sentence remains felicitous within her beliefs. Example (8) is more tricky. Apparently negation also creates a temporary context for interpretation. The infelicity prediction/ presuppositional inference at the over-all context is cancelled. Note the influence of the cue phrases 'so'; without it the example would remain infelicitous.

(8) John does not have a dog. So, it is not the case that *John's dog* is happy.
*pres:* —

I do realize this example sounds odd. Perhaps the oddity can be explained as follows. Although the presupposition is cancelled, the utterance does not add any new information. Therefore the utterance does not express a proper *assertion* and will be judged infelicitous[10].

## 3.5 global, local, intermediate

I suggested that sometimes interpretation shifts to a *temporarily context.* This approach was suggested by Heim (1983) and developed in (Van der Sandt, 1989; Zeevat, 1992). How does a temporary context arise? Take another look at the clause for negation in Definition 2. First we calculate, temporarily, the result of updating $\sigma$ with $\phi$. Then we subtract this from the original $\sigma$. Suppose $\phi$ contains a trigger. Now there are two versions of the context $\sigma$ that may be adjusted to accommodate the presupposition! The *local context*, inside the scope of the negation, or the both the local and the *global context*, outside its scope. Here is Example (8) again. Words are represented by the first character.

(8) Assume $\sigma \Vdash \neg d$

$$\sigma[\neg(\partial d \wedge h(d))] = \sigma \setminus \sigma[\partial d][h(d)] = undef \quad \text{(global)}$$

$$\sigma \setminus \sigma[d][h(d)] = \sigma \setminus \mathbb{0} = \sigma \quad \text{(local)}$$

[10]Compare the principles of assertion formulated by Stalnaker (1979)

This explains the possibility of cancelling in case of a negation. Apparently, in most cases, accommodation to the global context is preferred. But when there is a local context and global accommodation is not allowed, it is possible to accommodate only locally. Note that under this interpretation local accommodation must not result in undefinedness, but in the inconsistent state. When there is no temporary local context to accommodate, the utterance remains undefined.

Presuppositional inference, then, is the effect of presupposition accommodation at the original global context level. When assuming that all relevant information with respect to the presupposition is given, (i.e. using $\models^{\mathbb{1}}$) this form of inference indeed turns out to be non-monotonic.[11]

(9) $\neg(\partial(d) \wedge h(d)) \models^{\mathbb{1}} d$ but
$\neg d \wedge \neg(\partial(d) \wedge h(d)) \not\models^{\mathbb{1}} d$

So a simple consistency check (cautious accommodation), together with the update semantics notion of entailment, already produces the non-monotonic behavior that is characteristic of presupposition. Therefore, I believe, there is no need for a specific calculus of presuppositions based on *default logic*, e.g. (Mercer, 1992).

Conditionals lead to potentially three versions of the context to accommodate: the *local*, the *intermediate* and the *global* levels. Now it may be that one of the temporary contexts already supports the presupposition. In that case, there is no need for further accommodation: the presupposition is said to be *bound*. So a presupposition, like an anaphoric expression, may get bound to an antecedent (Van der Sandt, 1989).

(7) a. $b \to (\partial d \wedge h(d)) \models^{\mathbb{1}} d$ since

$$\begin{array}{lll} \sigma[b \to (\partial d \wedge h(d))] = & & \\ \sigma \setminus (\sigma[b] \setminus \sigma[b][\partial d][h(d)]) \Leftrightarrow & \sigma[d][b \to h(d)] & \text{(global)} \\ & \sigma[(d \wedge b) \to h(d)] & \text{(intermediate)} \\ & \sigma[b \to (d \wedge h(d))] & \text{(local)} \end{array}$$

b. $d \to (\partial d \wedge h(d)) \not\models^{\mathbb{1}} d$ since

$$\begin{array}{lll} \sigma[d \to (\partial d \wedge h(d))] = & & \\ \sigma \setminus (\sigma[d] \setminus \sigma[d][\partial d][h(d)]) \Leftrightarrow & \sigma[d \to h(d)] & \text{(bound)} \end{array}$$

### 3.6 Stacks

A conditional leads to three temporary versions of the context. One could imagine that a sentence like (10) leads to a *stack* of possible contexts for interpretation, corresponding to beliefs, indirect speech and regrets.[12]

[11] Compare Examples (5a) and (8). Proof by definition of $\models^{\mathbb{1}}$.

[12] Although this is one of the motivations of the stack model, I can't go into the details of modeling propositional attitudes. See (Hulstijn, 1995) for some suggestions.

(10) John believes that Mary said that he regrets being bald.

In fact, I believe, discourse structure looks like a tree. Contexts of interpretation are organized in some partial order that reflects the development of the conversation. However, looking from the current context towards the root of the tree, the structure is a stack of subsuming contexts.

Definition 3 (stacks)
*For $\Sigma$ a set of information states, define*

$$Stack_\Sigma \quad = \quad \{\epsilon\} \cup \{\langle \sigma, S\rangle \mid \sigma \in \Sigma, S \in Stack_\Sigma\}$$

We now need to lift update semantics to the stack level, defining how the logical operators interact with the stack structure.[13]

Definition 4 (update semantics - stacks)
*$US'$ is given by $\langle Stack_{\Sigma_{US}}, [.]'\rangle$ where*

$$\begin{aligned}
\langle \sigma, S\rangle[p]' &= \langle \sigma[p], S\rangle \quad (p \in \mathcal{A}) \\
\langle \sigma, S\rangle[\neg\phi]' &= \langle \tau' \setminus \tau, T\rangle \text{ where } \langle \tau, \langle \tau', T\rangle\rangle = \langle \sigma, \langle \sigma, S\rangle\rangle[\phi]' \\
\langle \sigma, S\rangle[\phi \wedge \psi]' &= S[\phi]'[\psi]' \\
\langle \sigma, S\rangle[\partial\phi]' &= \textit{depends on accommodation strategy}
\end{aligned}$$

Now that we have stacks, it becomes possible to define accommodation strategies that behave differently at different levels. Therefore a third local strategy makes sense. It embodies the idea that presupposition, just like asserted information, is part of the information content of a trigger.

Strategy 3 (content)

$$\sigma[\partial\phi] \quad = \quad \sigma[\phi]$$

Employed within stacks the *content strategy* makes it possible to model Russell's approach to presupposition and negation: a failing presupposition simply leads to falsity. The presupposition is projected to the global level, outside the scope of the negation. The resulting ambiguity of negation is modeled by the stack.

Strategy 4 (Russell)

$$\begin{aligned}
\langle \sigma, \epsilon\rangle[\partial\phi]' &= \langle \sigma[\partial_{content}\phi], \epsilon\rangle \\
\langle \sigma, S\rangle[\partial\phi]' &= \langle \sigma, S[\partial\phi]'\rangle
\end{aligned}$$

[13] The same definition can be formulated using standard stack operations *pop*, *push* and *top*. All stack operations behave strict with respect to undefinedness.

(11) It is not the case that the king of France is bald.
$\langle \sigma, \epsilon \rangle [\neg(\partial kof \wedge b(kof))]' =$
$\langle (\sigma[kof] \setminus \sigma[b(kof)]), \epsilon) \rangle \Leftrightarrow \langle \sigma, \epsilon \rangle [kof \wedge \neg b(kof))]'$

The following strategy comes closest to the ideas of Heim, as reconstructed by (Zeevat, 1992; Beaver, 1993). There is accommodation proper, but when the global context is incompatible, the resulting update is undefined. When an intermediate context already supports the presupposition (bound), it is not projected any further.

STRATEGY 5 (Heimian)

$$\begin{array}{lcl} \langle \sigma, \epsilon \rangle [\partial \phi]' & = & \langle \sigma[\partial_{cautious} \phi], \epsilon \rangle \\ \langle \sigma, S \rangle [\partial \phi]' & = & \left\{ \begin{array}{ll} \langle \sigma, S \rangle & if\, \sigma \Vdash \phi \\ \langle \sigma[\partial_{content} \phi], S[\partial \phi]' \rangle & otherwise \end{array} \right. \end{array}$$

This strategy covers most of the observations. It can't deal with all of the cancelling cases. In particular, Example (7e) and (8) would come out undefined. Example (8) is dodgy anyway. So, if only for the belief cases, that would seem a good result.

To be able to deal with the cancelling cases I propose the *full* strategy. It is based on the principle that presuppositions are accommodated along the stack, as close to the root as possible (Van der Sandt, 1989).

STRATEGY 6 (full)

$$\begin{array}{lcl} \langle \sigma, \epsilon \rangle [\partial \phi]' & = & \langle \sigma[\partial_{cautious}] \phi], \epsilon \rangle \\ \langle \sigma, S \rangle [\partial \phi]' & = & \left\{ \begin{array}{ll} \langle \sigma[\partial_{content} \phi], S[\partial \phi]' \rangle & if\, S[\partial \phi]' defined \\ \langle \sigma[\partial_{content} \phi], S \rangle & otherwise \end{array} \right. \end{array}$$

The principle that presuppositions always project as close to the root as possible, has been under attack. The following example sounds a bit odd: it seems to suggest that most German housewifes own a Porsche. That would suggest a local accommodation reading. However, the full accommodation strategy predicts intermediate accommodation, since global accommodation is not possible.

(12) Every German housewife washes *her Porshe* on Sundays.
$\sigma[g \rightarrow (\partial(p \wedge own(g,p)) \wedge wash\_on\_sunday(g,p))] \Leftrightarrow$
$\sigma[(g \wedge p \wedge own(g,p)) \rightarrow wash\_on\_sunday(g,p)]$ (intermediate)
$\sigma[g \rightarrow (p \wedge own(g,p) \wedge wash\_on\_sunday(g,p))]$ (local)

Therefore, the full strategy needs to be relaxed. The stack defines the set of potential accommodation contexts. Other factors, like pragmatic constraints or background knowledge determine the preferred location for accommodation among those. Contexts introduced by propositional attitudes may behave different from those introduced by negation. I realize that finding such constraints is by no means trivial. In this paper, I have only attempted to describe the general structure, needed to express the logical properties that were observed in the literature. Following Zeevat (1992) I take this structure to be a stack of information states. My contribution has been a precise formulation of the standard approaches from the literature. From a technical point of view, the full strategy, is most attractive. From an empirical point of view, the full strategy and the Heimian strategy can be distinguished only by their behavior with respect to negation and belief.

# 4 Implementation

The system has been implemented in Prolog. The system is designed to experiment with different accommodation strategies. Moreover, implementing the theory has uncovered hidden ambiguities. In this section I will briefly discuss some of the implementation details. Figure 1 gives the architecture of the system.

$$\begin{array}{ccccc}
 & & \textit{The man}\ \text{walks.} & & \\
 & & \Downarrow & & \\
 & & \boxed{\quad\text{Grammar}\quad} & \Longleftarrow & \text{lexicon} \\
 & & \Downarrow & & \\
 & & \partial(c_{21} \wedge man1(c_{21})) \wedge walk1(c_{21}) & & \\
 & & \Downarrow & & \\
\sigma & \Longrightarrow & \boxed{\quad\text{Update}\quad} & \Longrightarrow & \sigma' \\
\Updownarrow & & & & \Updownarrow \\
repr(\sigma) & \Longrightarrow & \boxed{\text{Constructive Update}} & \Longrightarrow & repr(\sigma')
\end{array}$$

Figure 1: architecture

The *grammar* module translates sentences of a fragment of English into $L$. I use the standard Prolog *Definite Clause Grammar* formalism (see Figure 2).

The *update* module calculates the result of updating the current information state or stack. So the most obvious way to implement the update module, is to find a representation of *information states*. The transitions between information states of the update module can then be mapped onto transitions between representations of information states.

Alternatively, we can put the maintenance of the representations of information states into the semantics directly. So utterances are now interpreted

```
s(S) --> np(VP^S),vp(VP).
np((C^VP)^(pres(C&N)&VP)) --> [the],n(C^N), { const(C) }.
np((C^VP)^(C&N&VP)) --> [a],n(C^N), { const(C) }.
np((C^VP)^(pres(C&VP)) --> [he], { const(C) }.
vp(X^walk1(X)) --> [walks].
n(X^man1(X)) --> [man].
```

Figure 2: sample DCG grammar

as transitions between representations. Update semantics models information states as subsets of a given set of possibilities $W$. By contrast, the representations need to be *constructed* incrementally. Therefore, this approach of defining the semantics of $L$ directly on constructed representations of information states, may be labeled a *constructive update semantics.*

## 5 Semantic Tableaux as Information States

*Semantic tableaux* form a sound and complete deduction method for both propositional and predicate calculus (Beth, 1959; Smullyan, 1968). I propose to use semantic tableaux to represent information states. The link between information states and semantic tableaux has been made before. See for instance (Landman, 1986) or work on logic-based knowledge representation (Elfrink and Reichgelt, 1989) .

A semantic tableau is a set of *branches.* A branch is a consistent set of *literals.* A literal is an atomic proposition or a negation of an atomic proposition. Tableaux are constructed from sequences of formulas using substitution rules. We start with formulas $\phi_1, \ldots, \phi_n$ and the initial tableau, $\{\{\}\}$. Rules are applied to the sequence of formulas. A double negation is eliminated. A conjunctive formula is replaced by both its conjuncts. A disjunctive formula results in a *split*: each disjunct is added to a separate copy of the current branch.[14] Tableau construction stops when all double negations, conjunctive formulas or disjunctive formulas have been dealt with: only branches of literals remain. Such a tableau is called a *complete* tableau. A branch is *closed* or eliminated from the tableau when it contains contradicting literals. A tableau is *closed* when all branches are closed. A branch $Lits_i$ on a

[14]In Definition 5 disjunction results in a split of the complete tableau. The original tableau and LeanTap both work on single branches. For propositional logic the approaches are logically equivalent.

complete tableau corresponds to a partial valuation $i$ as follows:

$$i(p) = \begin{cases} 1 & \text{iff } p \in Lits_i \\ 0 & \text{iff } \neg p \in Lits_i \\ undef & \text{otherwise} \end{cases}$$

A tableau is a way to enumerate all possible minimal valuations that satisfy the original formula. So when the complete semantic tableau for $\phi_1, \ldots, \phi_n$ closes, there is no valuation that satisfies $\phi_1, \ldots, \phi_n$. In other words, the sequence $\phi_1, \ldots, \phi_n$ is inconsistent.

The semantic tableau deduction method itself can be redefined as a *constructive update semantics.* The complete semantic tableau for a sequence $\phi_1, \ldots, \phi_n$ represents the $US$ information state $\mathbb{1}[\phi_1] \ldots [\phi_n]$. (by completeness) The substitution rules are interpreted as updates. Negation is modeled by so called *constructible falsity*: using the tableau rules, negations are projected down towards the literal level. Disjunction and implication are dealt with using the standard equivalences. Adding information to a tableau, means either extending or eliminating branches. Hence the new definition of $\sqsubseteq_{CUS}$. The initial information state $\mathbb{1}$ corresponds to the initial tableau $\{\{\}\}$. The inconsistent information state $\mathbb{0}$ corresponds to the closed tableau $\{\}$. Finally, a tableau is said to *support* a formula, when adding the negation of the formula would close it.[15]

Definition 5 (CUS)
*$CUS$ is given by $\langle \Sigma_{CUS}, [.]_{CUS} \rangle$ where*

$$\begin{aligned} \sigma[\neg\neg\phi]_{CUS} &= \sigma[\phi]_{CUS} \\ \sigma[p]_{CUS} &= \{Lits_i \cup \{p\} \mid i \in \sigma, \neg p \notin Lits_i\} \\ \sigma[\neg p]_{CUS} &= \{Lits_i \cup \{\neg p\} \mid i \in \sigma, p \notin Lits_i\} \\ \sigma[\phi \wedge \psi]_{CUS} &= \sigma[\phi]_{CUS}[\psi]_{CUS} \\ \sigma[\neg(\phi \wedge \psi)]_{CUS} &= \sigma[\neg\phi]_{CUS} \cup \sigma[\neg\psi]_{CUS} \end{aligned}$$

$$\begin{array}{llll} \sigma & \sqsubseteq_{CUS} & \tau & \textit{iff} \quad \forall j \in \tau, \exists i \in \sigma \textit{ such that } Lits_i \subseteq Lits_j \\ \sigma & \Vdash_{CUS} & \phi & \textit{iff} \quad \sigma[\neg\phi]_{CUS} = \mathbb{0} \end{array}$$

## 6 Preprocessing

It is not possible to directly add a clause for $\partial\phi$ to the definition of $CUS$. Because of *constructible falsity* the characteristic interaction of presuppositions and negation is lost. It would be possible to define an operation for

[15] An alternative definition of support is the following. $\sigma \Vdash^*_{CUS} \phi$ iff $\sigma[\phi]_{CUS} \sqsubseteq_{CUS} \sigma$. For propositional logic the notions are equivalent (Hulstijn, 1995). More research into the notions of support and entailment for $CUS$ is needed.

tableaux analogous to $\setminus$ on information states. Effectively a tableau is a disjunction of branches, each consisting of a conjunction of literals. The complement of a tableau would then be, using De Morgan's rules, a huge join of all possible complements of each branch. Calculating such a join would not be very efficient. Besides, we also need stacks to properly model the Heimian and full accommodation strategies.

Therefore, both the stack structure and negation will be simulated, using what I call a *preprocessing* implementation. Instead of pushing temporary information states onto the stack, I will push formulas that represent the temporary information states. Note that the representation uses no empty stacks. A stack consisting of a single information state is represented by $\langle\top, \sigma\rangle$.

Definition 6 (formula stacks)
*For $\Sigma$ a set of information states, $\phi \in L$ define*

$$FStack_\Sigma \quad = \quad \{\langle\phi, \sigma\rangle\} \cup \{\langle\phi, S\rangle \mid \sigma \in \Sigma, S \in FStack_\Sigma\}$$

This move, of pushing formulas instead of temporary contexts, becomes possible, because the definition of negation in $US$ can be altered without a change in the resulting semantics.[16]

$$\sigma[\neg\phi] \quad = \quad \sigma \setminus \mathbb{1}[\phi]$$

An additional advantage of this approach is that it becomes possible to use a standard theorem prover to implement the ideas behind $CUS$. I use LeanTap (1994). LeanTap is a very simple and elegant theorem prover, based on semantic tableaux. It is implemented in Prolog. I have adapted LeanTap in such a way that it explicitly stores the complete tableau for a sequence of formulas. Originally, storage only occurred at runtime, using the build-in prolog goal stack. It is easy to prove that the Prolog goal `leantap(Phi,S,S1)` succeeds if $\sigma[\phi]_{CUS} = \sigma'$ where `Phi,S,S1` represent $\phi, \sigma, \sigma'$ respectively.

Definition 7 (update semantics — preprocessing)
*The preprocessing implementation can be characterized by $\sigma[\phi]_{pre} = \tau$ if $\langle\top, \sigma\rangle[\phi]'' = \langle\phi', \sigma'\rangle$ and $\sigma'[\phi']_{CUS} = \tau$ where*

$$\begin{array}{rcl}
\langle\phi, S\rangle[p]'' & = & \langle\phi \wedge p, S\rangle \textit{ with } (p \in \mathcal{A}) \\
\langle\phi, S\rangle[\neg\psi]'' & = & \langle\phi' \wedge \neg\psi', S'\rangle \textit{ if } \langle\top, \langle\phi, S\rangle\rangle[\psi]'' = \langle\psi', \langle\phi', S'\rangle\rangle \\
\langle\phi, S\rangle[\psi \wedge \chi]'' & = & \langle\phi, S\rangle[\psi]''[\chi]'' \\
\langle\phi, S\rangle[\partial\psi]'' & = & \textit{depends on accommodation strategy}
\end{array}$$

[16]The reason there was a complete copy of the global context in the first place, was the possibility of accommodation. Now that is dealt with explicitly.

In order to complete the implementation, we need to reformulate the accommodation strategies for formula stacks. The local strategies, *classic*, *cautious* and *content* work on single information states. We can use $\Vdash_{CUS}$ and $[.]_{CUS}$ instead of the original $US$ notions.

The stack-based strategies, *Heimian* and *full* can be reformulated too. The global context (the end of the recursion) is now available as $\sigma$ in for instance $\langle \phi, \langle \phi', \sigma \rangle \rangle$. The local context is now modeled partly by a formula, partly by the formulas down the stack and partly by the global context. So for consistency checks or entailment the formula stack needs to be *flattened*: the global context is updated with all pushed formulas.

$$\begin{array}{rcl} flatten(\langle \phi, \sigma \rangle) & = & \sigma[\phi]_{CUS} \\ flatten(\langle \phi, S \rangle) & = & (flatten(S))[\phi]_{CUS} \end{array}$$

I realize this not an efficient way of encoding the effect of presupposition on the stack. A lot of equivalent states are recomputed every time. An slight improvement would be to store states for later usage. Remember that stack operations behave strict with respect to undefinedness. By way of example, here is the Heimian strategy for formula stacks.

Strategy 7 (Heimian″)

$$\begin{array}{rcl} \langle \phi, \sigma \rangle [\partial \psi]'' & = & \langle \phi, \sigma[\partial_{cautious} \psi] \rangle \\ \langle \phi, S \rangle [\partial \psi]'' & = & \left\{ \begin{array}{ll} \langle \phi, S \rangle & if\, flatten(\langle \phi, S \rangle) \Vdash_{CUS} \psi \\ \langle \phi \wedge \psi, S[\partial \psi]'' \rangle & otherwise \end{array} \right. \end{array}$$

## 7 Conclusions

A number of conclusions can be drawn from this study into the logical nature of presupposition accommodation.

The use of Update Semantics with stacks makes it possible to formalize existing standard approaches to presupposition accommodation in the same framework. Using this tool the empirical predictions of different approaches can be more easily compared.

Presuppositional inference can be explained by the process of presupposition accommodation to the global context. A simple consistency check and the Update Semantics notion of entailment already produce the characteristic non-monotonic behavior. No need for a fancy default logic.

The full strategy best models the standard observations discussed under *precondition*, *accommodation proper*, *presupposition test* and *cancelling*. Context introduced by beliefs may behave different from those introduced by negation. Constraints from domain knowledge may overrule the presupposition strategy.

Semantic tableaux can be used to represent information states. When negation is implemented by *constructible falsity*, we need to simulate the characteristic interaction between presupposition and negation. So called *formula-stacks* can be used to that effect. Using a *pre-processing* implementation with formula stacks, makes it possible to translate the accommodation strategies to the implementation directly.

A constructive semantics is one in which the meaning of an utterance is expressed as the transition between representations of the context, that are constructed incrementally. A constructive semantics might advance natural language understanding. For instance, the development of man-machine dialogue systems can be based on a constructive model of the information changes during the dialogue (Bunt, 1994). Other applications may be in information retrieval. The tableaux and stacks in this paper are somewhat misleading: they only represent the information content. But the way in which information is presented does matter. Word order, previously asked questions, intonation and grammatical constructions like topicalization, all influence the flow of a dialogue and therefore the construction of the context. Also extra-linguistic factors, such as social obligations, specific domain knowledge or the goal of the speaker are important. Therefore, $CUS$ is not a very useful constructive semantics.

## References

Beaver, David. 1993. The kinematics of presupposition. Technical Report R2.2.A part II, DYANA-2.

Beckert, Bernhard and Joachim Posegga. 1994. Leantap: Lean tableau-based deduction. Technical report, Universität Karlsruhe.

Beth, E. 1959. *The Foundation of mathematics.* North Holland, Amsterdam.

Bunt, H. C. 1994. Context and dialogue control. *Think*, 3:119–31.

Elfrink, Bernie and Han Reichgelt. 1989. Assertion-time inference in logic-based systems. In Peter Jackson, Han Reichgelt, and Frank van Harmelen, editors, *Logic Based Knowledge Representation.* MIT Cambridge Mass.

Gazdar, Gerald. 1979. *Pragmatics: Implicature, Presupposition and Logical Form.* Academic Press, New York.

Grice, H. P. 1975. Logic and conversation. In Jerry L. Morgan Peter Cole, editor, *Syntax and Semantics*, volume 3. Academic Press.

Heim, Irene. 1983. On the projection problem for presuppositions. WCCFL, 2.

Hulstijn, Joris. 1995. Presupposition in update semantics. Master's thesis, Faculty of Mathematics and Computer Science, University of Amsterdam.

Karttunen, Lauri. 1973. Presuppositions of compound sentences. *Linguistic Inquiry*, IV(2):169–193.

Karttunen, Lauri. 1974. Presupposition and linguistic context. *Theoretical Linguistics*, I:181–194.

Landman, Fred. 1986. Conflicting presuppositions and modal subordination. In *Papers from the 22nd meeting, Chicago Linguistic Society: 195:207.*

Lewis, David. 1979. Scorekeeping in a language game. *Journal of Philosophical Logic*, (9):339–359.

Mercer, R. 1992. Default logic and presupposititition. *Journal of Semantics*, 9(3).

Smullyan, R. M. 1968. *First-Order Logic.* Springer-Verlag, Berlin.

Stalnaker, Robert. 1979. Assertion. In Peter Cole, editor, *Pragmatics*, number 9 in Syntax and Semantics. Academic Press, New York, pages 315–32.

Van der Sandt, R. 1989. Presupposition and discourse structure. In *Semantics and Contextual Expression.* Foris, Dordrecht, pages 26–94.

Veltman, Frank. 1996. Defaults in update semantics. *Journal of philosophical logic*, 25(3):221–262. previous version in DYANA R2.5A, 1990.

Zeevat, Henk. 1992. Presupposition and accomodation in update semantics. *Journal of Semantics*, 9:379–412.