Logic and Control: How They Determine the Behaviour of Presuppositions

Marcus Kracht
II. Mathematisches Institut
Freie Universität Berlin
Arnimallee 3
14195 Berlin

1 Three Problems for Presuppositions

Presupposition is one of the most important phenomena of non-classical logic as concerns the applications in philosophy, linguistics and computer science. The literature on presuppositions in linguistics and analytic philosophy is rather rich (see [Sa 88] and the references therein), and there have been numerous attempts in philosophical logic to solve problems arising in in connection with presuppositions such as the projection problem. In this essay I will introduce a system of logics with control structure and elucidate the relation between context-change potential, presupposition projection and three-valued logic.

For a definition of what presuppositions are consider these three sentences.

(1) Hilary is not a bachelor.

(2) The present king of France is not bald.

(3) $\lim_{n\to\infty} a_n \neq 4$

Each of these sentences is negative and yet there is something that we can infer from them as well as from their positive counterparts; namely the following.

($1^\dagger$) Hilary is male.

($2^\dagger$) France has a king.

($3^\dagger$) $(a_n)_{n\in\mathbb{N}}$ is convergent.

This is impossible under classical circumstances. In classical logic, nothing of significance can be inferred from both P and $\neg P$ – but here we can infer non-trivial conclusions from both a sentence and its negation. Exactly how does this come about? The most popular answer has been given by Strawson. According to him a sentence may or may not *assert* something; the conditions under which a sentence asserts are not only syntactic but also *semantic* in nature. So, while *Dog the table very which under* is syntactically ill-formed and for that reason fails to assert, the sample sentences given above are syntactically well-formed and yet may fail to assert, namely when some conditions are not met. According to Strawson we say that a sentence S **presupposes** another sentence T if whenever S asserts, T is true. For example, (3) presupposes a_n *is convergent* since the former is assertive only if the latter is true. We will not question this view here; all we ask of the reader at this stage is his consent that the given intuitions are sound. If they are, sentences can no longer be equated with propositions. A sentence is a proposition only if it is assertive. Assertivity depends on the facts and hence it is not possible to say outright whether a given sentence is a proposition; this can vary from situation to situation. The distinction between sentences and propositions carries over to logic if we want to hold on to the assumption that sentences must have truth values. Then, as the classical truth-values shall continue to function in the same way, in a given model a proposition is still defined to be a sentence that is either true or false. On the grounds that there exist non-propositions we need to postulate at least one more truth-value, which, by penalty of self-contradiction, cannot mean *has no truth-value*; rather, it means *has no classical truth value.*

Three problems of presupposition theory can be isolated with which we will deal in turn. These are the *separation problem*, the *projection problem* and the *allocation problem.* The projection problem has a long intellectual history in linguistics. Intuitions have oscillated between an interpretation of presuppositions as a reflex of logic or as a result from the speaker-hearer-interaction, in short between a purely semantic and a pragmatic account. Strong Russellianists such as [Ke 75] want to deny it any status in semantics while most semanticists try to derive as many of the projection phenomena from their theory as they can. We will see shortly that both must be wrong. If semantics has to do with meaning that is static (at least in the short run) then computer languages and mathematical jargon provide solid evidence that three valued logic is here to stay and presupposition has a home in semantics. Yet, if one and the same sentence has two different meanings, that is, if we acknowledge that there are cases of ambiguity with respect to the presuppositions which are only resolved by the context, there must be more to presupposition than a semantical theory can provide. The problem of projection is generally stated as follows. Suppose that a sentence S is built from some simple sentences $S_1, \ldots, S_n$ and that we know the presuppositions of $S_1, \ldots, S_n$, can we compute the presuppositions of S? This is really a non-trivial question. A first inspection of examples suggests that presuppositions are simply accumulated; this is the theory of [La 71]. But consider (4).

(4) If $(a_n)_{n\in\mathbb{N}}$ is convergent then $\lim_{n\to\infty} a_n \neq 4$.

It has quickly been found that if S presupposes T then T *and* S as well as *If* T *then* S do not presuppose T. So there is a general question as to why this is so and what other rules of projection are valid. It has gone unnoticed that the way in which we have stated the projection problem it becomes ambiguous or at least hopelessly untractable in view of the data that has been accumulated over the years. It is known, namely, that the logical form of a complex sentence need not directly conform to the logical form of the message communicated. Subordination provides one example; another, rather vexing example is the following announcement that could be seen e. g. in a cinema. (This example is due to [Ra 79].)

(5) Old age persons $\left\{ \begin{array}{c} \textit{and} \\ \textit{or} \end{array} \right\}$ students at half price.

No matter whether the board says *and* or *or*, we read the same message out of it. This means that we have to postulate two levels of representation similar to syntax: the surface form, called here *syntactic logical form*, and the logical form of the message that is communicated. We use the word *message* rather loosely here but it should be clear that it is not the same as the utterance of the sentence or meaning thereof. The logical form of the message will be called the *underlying* (or *semantic*) *logical form*. The map from the syntactic logical form to the underlying logical form is not unique and it is not at all clear how the two relate; to spell this out in detail is the the *problem of the underlying logical form*. Once the underlying logical form is found, the question how the presupposition of a complex expression is computed can be asked again and answers can be given by direct calculation; we will show how this is done using three valued logic. It is in the following sense that I will understand the projection problem: Given the underlying logical form of a sentence, what are its presuppositions?

Distinct from the projection problem in the narrow sense is the *allocation problem*. [1] To formulate it we assume that the semantics of words or phrases gives rise to explicit presuppositions. Though there is no surface connective that is equivalent to the presuppositionification operators $\downharpoonleft$ or ∇ to be defined later, there are constructions or words that need to be translated using $\downharpoonleft$ or ∇. Typical examples are *know* or *bachelor*. In DRT terms, they create a special presuppositional box which I call a *semantic anchor*. This anchor needs to be dropped somwhere. That there really is a choice between places at which to drop this anchor, let us consider the next examples.

(6) Everytime X saw four aces in Y's hand he signalled to *his partner*.

(7) If the judges make a mistake in the formal procedure *the lawyer* will persuade *his client* to appeal.

In (6), we can assume X to have a fixed partner, that is, we can read into (6) a *stronger statement than the one given*. (I should excuse myself here for not being precise; of course, by *statement given* I mean something like the syntactic logical form but this is not 100% right.) But we need not; if we assume that X is playing in each game with a different partner, the referent of *his partner* will depend on the chosen occasion in which X sees four aces in Y's hand. Similarly with (7). If the presupposition initiated by the phrase *the lawyer* stays local, (7) presupposes ($7^\dagger$). If it is chosen to be global, (7) presupposes ($7^\ddagger$).

($7^\dagger$) If the judges make a mistake in the formal procedure there is one and only one lawyer.

($7^\ddagger$) There is a unique lawyer.

Admittedly, ($7^\dagger$) sounds artificial because we are more inclined to say that the lawyer is sufficiently determined by the legal procedure and not the additional formal mistakes that may occur. We will see that this view is not justified; but even if it were this only fortifies our arguments that projection and allocation should be seen in the context of the problem of logical form. To account for these differences, we introduce a distinction between the

[1] I will later challenge the picture on presupposition allocation that I will now draw. For the moment it suffices that the problem itself becmoes clear.

origin of a presupposition and its *locus*. First we fix an underlying logical form with respect to the language, that is, we do not yet interpret the words by a semantical meta- or infra-language; we can assume this level to be some variant of LF in Government and Binding. Once we have done that we unpack the meaning of the occurring words in terms of the representational language, a process which accidentally also produces instances of presupposition creating operators ($\downharpoonright$ or ∇). Or, to use the DRT metaphor, the anchors are now being dropped. This is a straightforward translation; we call the place in the syntactic parse of the resulting translation at which a presupposition is created the **origin**. Mostly, the origin can be located in the original sentences by pointing at the item creating the presupposition. The representation thus created is only a transient one. For we now consider the question whether the presupposition should be placed somewhere else in order to obtain the correct underlying logical form. This 'placing somewhere else' can be understood as a kind of movement transformation that will remove the presupposition from its origin and reinsert it somewhere else. In order not to offend established associations I refer to this process as *reallocation*. The place at which a presupposition is finally placed is called the **locus**. [Sa 90] can be understood as a theory of allocation in our sense.

The **separation problem** originates from a distinction between *the* assertion of a sentence and *the* presupposition of a sentence. It consists in the problem to find, given a sentence S, two propositions (!) A and P such that

(sep) S is true iff A is true
S is a proposition iff P is true

The separation problem has received little attention; to our knowledge it has never been explicitly formulated. However, it is quite an important one since normal semantic theories assume that all their predicates used in the representation are bivalent. So, when *bachelor* finally receives an interpretation via, say, Montague translation, as $\mathbf{male}'(x)\wedge\mathbf{human}'(x)\wedge \mathbf{adult}'(x)\wedge\mathbf{unmarried}'(x)$ it is assumed that $\mathbf{male}'(x)$, $\mathbf{human}'(x)$, $\mathbf{adult}'(x)$ as well as $\mathbf{unmarried}'(x)$ are classical and can therefore be manipulated on the basis of a distinction between truth and falsity only. Many projection algorithms are defective in the sense that it they tacitly assume that they manipulate only propositions. In ordinary language, however, it is not at all clear that separation can be fully carried out. For even though we can name *sentences* that fulfill (sep) it is not clear whether we can have *propositions* to fulfill (sep). This is due to the fact that all predicates and operators in language are *typed*, that is, they need as input an object of a certain type in order to be well-formed. For example, $\mathbf{unmarried}'(x)$ requires an x that is human; otherwise *Consciousness is unmarried.* would count as a proposition.

Under limited circumstances, however, separation can be worked out to the bottom. Such circumstances are provided in mathematics. For example, $\lim_{n\to\infty} a_n \neq 4$ can be separated into

A : No density point of $(a_n)_{n\in\mathbb{N}}$ is equal to 4.
P : $(a_n)_{n\in\mathbb{N}}$ is bounded and has exactly one density point.

Likewise, $a/x = 6$ can be separated into

$A : a = 6x$
$P : x \neq 0$

The projection problem may also be formulated as follows. Given a sentence S composed from simple sentences $S_1, \ldots, S_n$; how to separate S on the basis of a separation $A_1 : P_1, \ldots, A_n : P_n$?

2 Some Notions from Logic

Propositional languages have the advantage of knowing only one type of well-formed expression, that of a proposition. Since there is a clash with the philosophical terminology I will refer to the propositions of an arbitrary propositional language as *terms*. Terms are produced from variables and connectives in the known way. Terms are interpreted in *algebras*. A propositional language defines a similarity type of algebras in which we can interpret the variables and also the terms. Let us fix such a language and call it $\mathbb{L}$. A *logic* over $\mathbb{L}$ is defined via a set of *rules* in the obvious way. A rule is a pair $\langle \Delta, Q \rangle$ where Δ is a finite set of terms called the *premisses* and Q a single term called the *conclusion*. We will not spell out the details here and refer instead to [Wó 88]. Logics correspond one-to-one with certain classes of *matrices*. A **matrix** is a pair $\mathfrak{M} = \langle \mathfrak{A}, D \rangle$ where $\mathfrak{A}$ is an algebra of the similarity type of $\mathbb{L}$ and D a set of elements of $\mathfrak{A}$. D is the set of *designated elements* of $\mathfrak{M}$. The rule $\langle \Delta, Q \rangle$ is *valid* in $\mathfrak{M}$ is for all valuations β we have $\beta(Q) \in D$ if only $\beta(\Delta) \subseteq D$. We write $\Delta \vdash_{\mathfrak{M}} Q$.

It is possible to give an analogical treatment of presupposition. In addition to designated truth values we need a distinction between **admitted** and **non-admitted** or **unwanted** truth-values. Technically, if we want to incorporate presupposition into logic we have to expand logical matrices by a set that tells us which truth-values are unwanted, just as we need a set of designated truth-values to tell us what truth is. It seems natural to say that designated truth-values are admitted and hence we get the following definition.

Definition 2.1 *A* ***p-matrix*** *or* ***presuppositional matrix*** *is a triple* $\mathfrak{P} = \langle \mathfrak{A}, D, U \rangle$ *where* $D \subseteq A$ *is a set of designated elements and* $U \subseteq A$ *a set of unwanted truth values, and moreover* $D \cap U = \emptyset$. *We say that* P ***presupposes*** Q ***relative to*** $\mathfrak{P}$ *– in symbols* $P \triangleright_{\mathfrak{P}} Q$ *– if for all valuations* β $\beta(P) \notin U$ *implies* $\beta(Q) \in D$.

A general theory of presuppositions in arbitrary languages is possible but we prefer to concentrate on three valued logic in relation to the three main issues of presuppositional theory. This does by no means imply that the abstract approach sketched here serves no real purpose. Indeed, in computer science there are more than one recognizable type of unwanted truth value, namely loop and fail. Moreover, in more sophisticated logics for natural language there sometimes is a need to have more than two truth-values. In all of these cases, a presuppositional theory can be added on top using these abstract methods.

3 Connectives with Explicit Control Structure

Unlike classical logic, three-valued logic forces us to think quite seriously about the meaning of simple connectives such as *and* and *or*. Indeed, there is no single best choice of a three-valued interpretation. Rather than arguing for one interpretation that it is best we will try to develop an understanding of the difference between these options. We will use a computational interpretation which has its origin in the discussion of [Ha 75]. At the heart of this interpretation lies a consistent reading of the third truth-value as computational failure (fail or, in our context U). This failure arises from improper use of partial predicates or functions e. g. dividing by 0, taking the square root of negative numbers etc. A second component is the addition of an explicit *control structure* that determines the actual computation of the truth-value. So, rather than using logic as a meta-language describing facts, we are interested now in a particular internal realization of logic, be it in a computer or in a human. The fundamental difference is that truth values are not immediately given to us just because they apply by logical force to the terms but we have to calculate in each case which term has which truth value. This makes no difference with respect to classical logic. But the fact that computations may fail and that this failure itself is counted as a truth-value intertwines logic with its implementation. To take a concrete example consider the following part of a Pascal program.

(8) $\mathsf{if}\ x < 0\ \mathsf{then}\ 1/(1-x) > 1+x;$

The second clause aborts if $x = 1$. However, the computer will never notice this, since the consequent is only considered if the antecedent is true; and the condition in the antecedent preempts this failure. In total, this sentence has no presupposition as far as the computer is concerned because it will under no circumstances fail. This, however, is due to two reasons. (1) The computer considers the consequent after the antecedent. (2) The computer drops the computation of the consequent in case the antecedent is not true. We could – in priciple – think of another strategy by which the consequent is checked first and the computation of the antecedent is dropped if the consequent is true. Then (8) will fail just in case $x = 1$. On the other hand, if the computer computes antecedent-to-consequent but looks at the consequent regardless of the antecedent, still it will fail if $x = 1$. So, both (1) and (2) are necessary.

We have isolated two properties of the standard computer implementations that produce the presuppositional behaviour of computers. One is the *directionality of computation* and the second is the *principle of economic computation*. If the computer implements no economy strategy the resulting logic is the so-called **Weak Kleene Logic** or **Bochvar's Logic**. It is characterized by the fact that any failure during a computation wherever it may arise will let the overall computation fail. The same logic will be derived even with the economy principle but with a different control structure. We can isolate four control structures; the first two are the uni-directional control structures *left-to-right* and *right-to-left*. The second are the bidirectional control structures; here, both directions are tried; the difference is whether the computation succeeds if only one branch succeeds (strong) or if both succeed (weak). These four cases correspond to four diacritics; $\overset{\triangleright}{\to}$ for left-to-right, $\overset{\triangleleft}{\to}$ for right-to-left, $\overset{\shortmid}{\to}$ for bidirectional and weak and $\overset{\diamond}{\to}$ for bidirectional and strong. The reader may check that this gives the following truth tables.

$\overset{\shortmid}{\to}$	T	F	U
T	T	F	U
F	T	T	U
U	U	U	U

$\overset{\triangleright}{\to}$	T	F	U
T	T	F	U
F	T	T	T
U	U	U	U

$\overset{\triangleleft}{\to}$	T	F	U
T	T	F	U
F	T	T	U
U	T	U	U

$\overset{\diamond}{\to}$	T	F	U
T	T	F	U
F	T	T	T
U	T	U	U

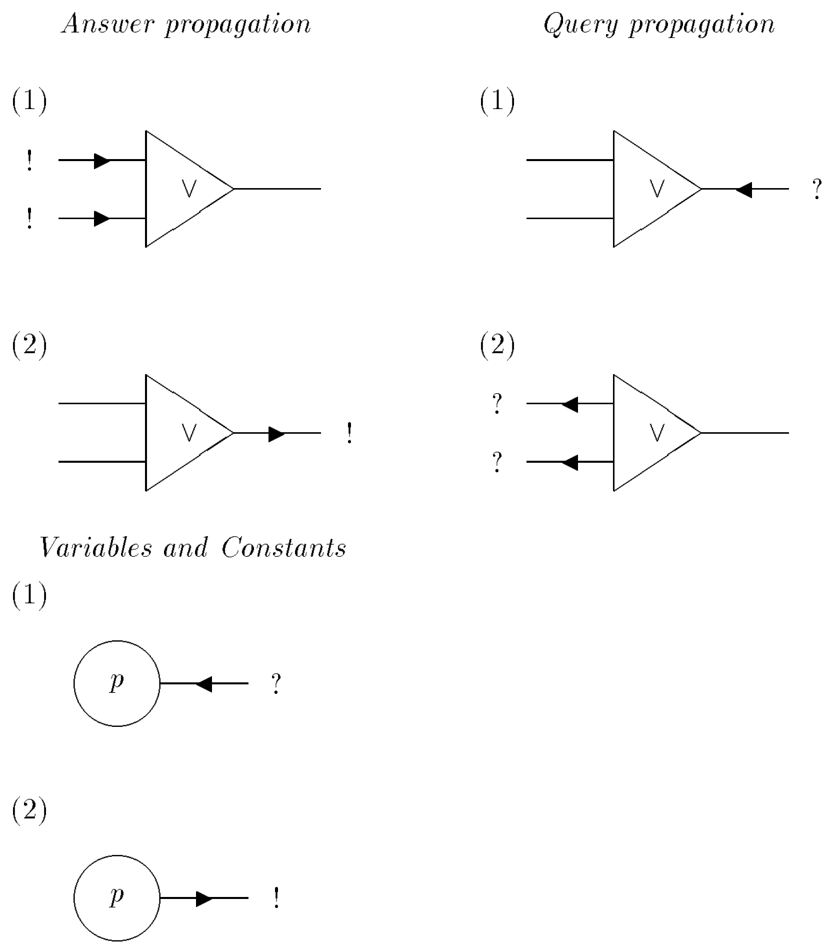
If we define the assignment relation $\leq$ between truth values in the obvious way ($\mathsf{U} \leq \mathsf{F},\mathsf{T}$) then the bidirectional connectives are defined from the unidirectional ones in the following way.

$$P \overset{\shortmid}{\to} Q = \min_{\leq}\{P \overset{\triangleright}{\to} Q, P \overset{\triangleleft}{\to} Q\}$$

$$P \overset{\diamond}{\to} Q = \max_{\leq}\{P \overset{\triangleright}{\to} Q, P \overset{\triangleleft}{\to} Q\}$$

There is also an interpretation that does not assume that computations may fail, i. e. that the basic predicates are partial, but nevertheless introduces three valued logic because it admits the possibility of broken channels in information transmission. Here we assume the connectives to be machines in a network which are supposed to answer queries. Once activated, the machines work the query backwards to the input channels. A single query can start an avalanche of queries which terminates in the variables. The latter we also regard as machines, working on no input; they are able to respond directly to a query. They can in principle give two answers, namely T and F. In that case, everything works as in classical logic.

Now suppose that the machines can also fail to respond because they are broken, because the connection is interrupted or because some machine fails to answer in due time. Our automaton $\vee$ somewhere in the network is thus faced with several options when the answers to the queries may turn out to be incomplete or missing at a time point. (a) It can wait until it receives the proper input; (b) It can use some higher order reasoning to continue in spite of an incomplete answer. (b) is the option with inbuilt economy principles and (a) is the option without. The (b) option branches into several distinct options. They are brought together as follows. We understand that in a binary function $f(\ _1,\ _2)$ there is an **information lock** between the first slot and the second slot. This lock has four positions. It can be closed ($\shortmid$), completely open ($\diamond$) and half-open, either to the right ($\triangleright$) or to the left ($\triangleleft$). These four positions determine in which way information about a received input i. e. about the value of the argument that is plugged in may flow. In closed position the left hand does not know what the right hand is doing. Even though P is received as T it is not known to the automaton when working at Q that P is true and it can therefore not know that it may now forget about the value of Q. It is still waiting. The same



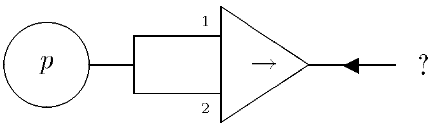
These pictures show the behaviour of logical automata. A connective may either compute an output answer (2) from the input answers (1) or an input query (2) from an output query. Variables transform output queries into output answers.

occurs if the information lock is open from right to left. The truth-values coincide with those given above. Of course we must now be careful with the boolean laws since it is not guaranteed that they hold. But the typical interdefinability laws of boolean logic hold for the connectives with similar control structure. We show some of them in the next lemma.

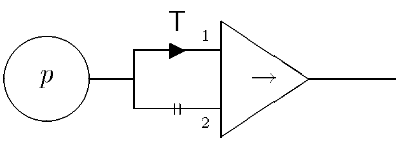
Lemma 3.1 *The following interdefinability laws hold:*

$$\begin{array}{rclcrcl}
P \overset{|}{\vee} Q & = & \neg(\neg P \overset{|}{\wedge} \neg Q) & \qquad & P \overset{|}{\rightarrow} Q & = & \neg P \overset{|}{\vee} Q \\
P \overset{\triangleright}{\vee} Q & = & \neg(\neg P \overset{\triangleright}{\wedge} \neg Q) & & P \overset{\triangleright}{\rightarrow} Q & = & \neg P \overset{\triangleright}{\vee} Q \\
P \overset{\triangleleft}{\vee} Q & = & \neg(\neg P \overset{\triangleleft}{\wedge} \neg Q) & & P \overset{\triangleleft}{\rightarrow} Q & = & \neg P \overset{\triangleleft}{\vee} Q \\
P \overset{\diamond}{\vee} Q & = & \neg(\neg P \overset{\diamond}{\wedge} \neg Q) & & P \overset{\diamond}{\rightarrow} Q & = & \neg P \overset{\diamond}{\vee} Q. \quad \blacksquare
\end{array}$$

It is instructive to see why this interpretation is sound even when the variables are assumed to be classical. The answer is not straightforward but simple. It is best understood with an example. We take $p \overset{\diamond}{\rightarrow} p$.



When a query is received, our automaton sends a query with both input channels. Suppose it receives answers as follows.



If we assume that the connection from 'p' to the second (= lower) input channel is broken, only the first channel receives the answer given by p. Yet the automaton cannot act. The reason is that the automaton does not know that the two input channels give the same answer; it only sees that they go out into the network but has no idea that they are systematically connected. So it is forced to wait for the second input. $p \overset{\diamond}{\rightarrow} p$ is thus not always true. The situation is comparable to that of a scientist concerned with the truth of two propositions r and s. Suppose that he does not know that they are in fact the same such as *Hesperus rises in the morning* and *Phosphorus rises in the morning.* So he cannot conclude that $r \stackrel{\diamond}{\rightarrow} s$ is true; in fact, only by long term experiment and/or statistical reasoning he may come to the conclusion that the two must be the same because the one is true exactly when the other is. Indeed, his local knowledge of the world compares directly to the limited knowledge of the $\stackrel{\triangleright}{\rightarrow}$-automaton.

Returning now to the issue of presupposition we notice that the explicit control structures give us the desired systematization of the logics; choosing uniformly the weak bidirectional connectives realizes Bochvar's Logic or Weak Kleene Logic, choosing the strong bidirectional connectives we get the Strong Kleene Logic and choosing the left-to-right interpretation gives us the typical computer implementations. In natural language the facts are not that simple. On closer inspection the connectives only show a certain tendency to be asymmetrical and left-to-right. But this default choice can be overridden as the bathroom sentences show.

(9) Either the bathroom is upstairs or there is no bathroom.

(10) If the bathroom is not upstairs there is none in the house.

It turns out that *or* tends to be weak and is directional only if there is a chance of cancelling a presupposition that would otherwise be inherited. Furthermore, *and* can be weak. Only *if ... then* shows a rather strong left-to-right tendency. But these are only rules of thumb.

4 A Formal Approach to the Projection Problem

In three-valued logic we agreed to let P be a proposition if $\beta(P) \in \{\mathsf{T}, \mathsf{F}\}$. So we have $D = \{\mathsf{T}\}$ and $U = \{\mathsf{U}\}$. An immediate consequence is that $P \triangleright Q$ iff $P \vee \neg P \vdash_3 Q$. For $P \vee \neg P$ is true iff P is either true or false, regardless of which of the four instantiations we choose. Thus if $P \triangleright Q$ then if $P \vee \neg P$ is true, P is either true or false and thus Q is true as well. And conversely. The same holds for classical logic. But since in classical logic no term can fail to be a proposition under no matter what valuation, the notion of presupposition becomes trivial.

Proposition 4.1 *In classical logic, P presupposes Q iff Q is a tautology.*

Proof. By definition, $P \triangleright_{\mathbf{2}} Q$ if Q is true whenever P is a proposition. Thus Q is always true, hence a tautology. ■

Only when we admit a third value the notion of presupposition starts to make sense. In the sequel we will indeed study presupposition in the context of three valued logic. We write $\triangleright$ for $\triangleright_{\mathbf{3}}$.

Proposition 4.2 *(0) $P \triangleright Q$ iff $P \vee \neg P \vdash_3 Q$.* ($\vee \in \{\overset{\shortmid}{\vee}, \overset{\triangleright}{\vee}, \overset{\triangleleft}{\vee}, \overset{\diamond}{\vee}\}$.)

(i) If $P \triangleright Q$ and $P \equiv_3 P', Q \vdash_3 Q'$ then $P' \triangleright Q'$.

(ii) If $P \triangleright Q$ and $Q \triangleright R$ then $P \triangleright R$.

(iii) $P \triangleright Q$ iff $\neg P \triangleright Q$.

(iv) If $P \triangleright Q$ and $Q \triangleright P$ then $P \equiv_3 Q$ and P (as well as Q) is not falsifiable.

Proof. (0) $\beta(P) \in \{\mathsf{T}, \mathsf{F}\}$ iff $\beta(P \vee \neg P) = \mathsf{T}$. Hence if $P \vee \neg P$ is true, P is true or false and by $P \triangleright Q$, Q is true. Also if $P \vee \neg P \vdash_3 Q$ and P is true or false, $P \vee \neg P$ is true and so Q is true. (i) Let $P \triangleright Q$, $P' \equiv_3 P$ and $Q \vdash_3 Q'$. Assume $\beta(P') \in \{\mathsf{T}, \mathsf{F}\}$. Then $\beta(P) \in \{\mathsf{T}, \mathsf{F}\}$ as well. Thus $\beta(Q) = \mathsf{T}$; now by $Q \vdash_3 Q'$ also $\beta(Q') = \mathsf{T}$. (ii) If $Q \triangleright R$ then a fortiori $Q \vdash_3 R$ and by (i) $P \triangleright R$. (iii) $P \triangleright Q$ iff $P \overset{\diamond}{\vee} \neg P \vdash_3 Q$ iff $\neg\neg P \overset{\diamond}{\vee} \neg P \vdash_3 Q$ iff $\neg P \triangleright Q$. (iv) Clearly, $P \triangleright Q$ implies $P \vdash_3 Q$ and $\neg P \vdash_3 Q$ and $Q \triangleright P$ implies $Q \vdash_3 P, \neg Q \vdash_3 P$. Thus $P \vdash_\mathbf{3} Q \vdash_\mathbf{3} P$. Furthermore, suppose that P is false; then Q is true and so P must be true as well. Contradiction. Similarly, Q cannot be false. Finally, if $\beta(P) = \mathsf{U}$, then Q cannot be true, otherwise P is true. Q is not false either, hence $\beta(Q) = \mathsf{U}$ as well. Dually, if $\beta(Q) = \mathsf{U}$ then also $\beta(P) = \mathsf{U}$. This shows $P \equiv_\mathbf{3} Q$. ■

Hence, if we consider the set $\mathfrak{N}$ of all nonfalsifiable formulae then $\triangleright$ turns out to be irreflexive and transitive on $\mathfrak{N}$. Let us now analyse the formal behaviour of presuppositions in languages of three valued logic. To this end we define a binary connective $\downharpoonleft$ by

$\downharpoonleft$	T	F	U
T	T	F	U
F	U	U	U
U	U	U	U

Proposition 4.3 $(Q \downharpoonleft P) \triangleright Q$. *Moreover, $P \triangleright Q$ iff $Q \downharpoonleft P \equiv_3 P$.*

Proof. The first claim is easy to verify. We turn to the second. ($\Rightarrow$) If $Q \downharpoonleft P$ is true then P is true. If P is true then by $P \triangleright Q$ also Q is true, hence $Q \downharpoonleft P$ is true. If $Q \downharpoonleft P$ is false, P is false. If P is false, Q is true by $P \triangleright Q$ and so $Q \downharpoonleft P$ is false. ($\Leftarrow$) If P is true, $Q \downharpoonleft P$ is true, so Q is true. This shows $P \vdash_3 Q$. If P is false then $Q \downharpoonleft P$ is false as well; and so Q is true showing $\neg P \vdash_3 Q$. So, $P \triangleright Q$. ■

It is possible to define a unary connective ∇ by $\nabla P := P \downharpoonleft P$. It has the following truth-table.

	∇
T	T
F	U
U	U

The idea of such a connective is due to D. Beaver. This unary connective allows to define presuppositions in a similar way as $\downharpoonleft$. This due to the fact established in the next theorem.

Proposition 4.4 $Q \downharpoonleft P \equiv_3 \nabla Q \overset{\shortmid}{\wedge} P \equiv_3 \nabla Q \overset{\triangleright}{\wedge} P$.

Proof. The formulae are truth-equivalent. For all three are true iff both P and Q are true. $Q \downharpoonleft P$ is false iff P is false and Q is true iff ∇Q is true and P is false iff $\nabla Q \overset{\shortmid}{\wedge} P$ is false iff $\nabla Q \overset{\triangleright}{\wedge} P$ is false. ■

To be precise: as soon as $\overset{\shortmid}{\wedge}$ or $\overset{\triangleright}{\wedge}$ or $\overset{\triangleleft}{\wedge}$ are definable, $\downharpoonleft$ is definable from ∇ and one of the three. It is therefore a matter of convenience whether we use $Q \downharpoonleft P$ or some definition using ∇.

Lemma 4.5 *Let* $*, \circ \in \{\overset{\shortmid}{\wedge}, \overset{\triangleright}{\wedge}, \overset{\triangleleft}{\wedge}, \overset{\diamond}{\wedge}\}$. *Then* $\nabla P * \nabla Q \equiv_3 \nabla(P \circ Q)$.

Proof. $\nabla P * \nabla Q$ can never be false since ∇P, ∇Q can never be false. Hence, we need to check only equivalence in truth. But $\nabla P * \nabla Q$ is true iff ∇P, ∇Q are true iff P and Q are true iff $P \circ Q$ is true iff $\nabla(P \circ Q)$ is true. ■

Lemma 4.6 $R \downharpoonleft (Q \downharpoonleft P) \equiv_3 (R \downharpoonleft Q) \downharpoonleft P$. *Furthermore, if* $* \in \{\overset{\shortmid}{\wedge}, \overset{\triangleright}{\wedge}, \overset{\triangleleft}{\wedge}, \overset{\diamond}{\wedge}\}$ *then* $R \downharpoonleft (Q \downharpoonleft P) \equiv_3 (Q * R) \downharpoonleft P$.

Proof. We show the second claim first. $R \downharpoonleft (Q \downharpoonleft P) \equiv_3 \nabla R \overset{\shortmid}{\wedge} (Q \downharpoonleft P) \equiv_3 \nabla R \overset{\shortmid}{\wedge} \nabla Q \overset{\shortmid}{\wedge} P \equiv_3 \nabla(Q * R) \overset{\shortmid}{\wedge} P \equiv_3 (R * Q) \downharpoonleft P$. Now for the first. $R \downharpoonleft (Q \downharpoonleft P) \equiv_3 \nabla R \overset{\shortmid}{\wedge} (Q \downharpoonleft P) \equiv_3 \nabla R \overset{\shortmid}{\wedge} (\nabla Q \overset{\shortmid}{\wedge} P) \equiv_3 \nabla(\nabla R \overset{\shortmid}{\wedge} Q) \overset{\shortmid}{\wedge} P \equiv_3 (R \downharpoonleft Q) \downharpoonleft P$. ■

The main theorem of this paragraph deals with the projection problem. Our solution is completely formal and bears only on the logical properties of presuppositions. We will approach the projection problem via *normal forms*.

Definition 4.7 *Let* $\mathbb{L}$ *be a language of boolean connectives with locks and* $\mathbb{L}^{\downharpoonleft}$ *its expansion by* $\downharpoonleft$. *We say a formula* $P \in \mathbb{L}^{\downharpoonleft}$ *is in* ***presuppositional normal form (pnf)*** *if* P *is syntactically equal to* $Q_2 \downharpoonleft Q_1$ *for some* $Q_{1,}, Q_2 \in \mathbb{L}$.

Theorem 4.8 *For each* $P \in \mathbb{B}_3$ *there exists a* $\pi(P) \equiv_3 P$ *which is in presuppositional normal form.*

Proof. The following are valid statements:

$$
\begin{array}{lll}
(as\downharpoonleft) & (R\downharpoonleft Q)\downharpoonleft P & \equiv_3 (R \wedge Q)\downharpoonleft P \\
(di\downharpoonleft) & R\downharpoonleft(Q\downharpoonleft P) & \equiv_3 (R \wedge Q)\downharpoonleft P \\
(ne\downharpoonleft) & \neg(Q\downharpoonleft P) & \equiv_3 Q\downharpoonleft(\neg P) \\
(co\downharpoonleft) & (Q\downharpoonleft P) \overset{|}{\wedge} R & \equiv_3 Q\downharpoonleft(P \overset{|}{\wedge} R) \\
 & (Q\downharpoonleft P) \overset{\triangleright}{\wedge} R & \equiv_3 Q\downharpoonleft(P \overset{\triangleright}{\wedge} R) \\
 & (Q\downharpoonleft P) \overset{\triangleleft}{\wedge} R & \equiv_3 (R \overset{\triangleright}{\rightarrow} Q)\downharpoonleft(P \overset{\triangleleft}{\wedge} R) \\
 & (Q\downharpoonleft P) \overset{\diamond}{\wedge} R & \equiv_3 (R \overset{\diamond}{\rightarrow} Q)\downharpoonleft(P \overset{\diamond}{\wedge} R)
\end{array}
$$

These statements if read from left to right, provide an algorithm for deriving a pnf for a formula containing negation and conjunctions. With the fact that all boolean connectives with locks can be represented with just negation and the conjunctions with locks, we can effectively reduce all terms by first eliminating the other booleans and then reducing according to the equivalences shown above. The task is thus to prove them. Lemma 4.6 showed $(as\downharpoonleft)$ and $(di\downharpoonleft)$, $(ne\downharpoonleft)$ is not difficult. For the laws $(co\downharpoonleft)$ observe the fact that all left hand sides and all right hand sides are true iff P, Q and R are true. So it is enough to establish that they receive U (or F) under the same valuations. **(1)** $(Q\downharpoonleft P) \overset{|}{\wedge} R$ is U iff $Q\downharpoonleft P$ is U or R is U iff either Q is not T or P is U or R is U iff Q is not T or $P \overset{|}{\wedge} R$ is U iff $Q\downharpoonleft(P \overset{|}{\wedge} R)$ is U. **(2)** $(Q\downharpoonleft P) \overset{\triangleright}{\wedge} R$ is U iff either $Q\downharpoonleft P$ is U or $Q\downharpoonleft P$ is T and R is U iff either Q is not T or P is U or P, Q are T and R is U iff Q is not T or $P \overset{\triangleright}{\wedge} R$ is U iff $Q\downharpoonleft(P \overset{\triangleright}{\wedge} R)$ is U. **(3)** $(Q\downharpoonleft P) \overset{\triangleleft}{\wedge} R$ is U iff R is U or R is T and $Q\downharpoonleft P$ is U iff R is U or R is T and: either P is U or Q is not T iff either R is U or R is T and P is U or R is T and Q is not T; $(R \overset{\triangleright}{\rightarrow} Q)\downharpoonleft(P \overset{\triangleleft}{\wedge} R)$ is U iff $R \overset{\triangleright}{\rightarrow} Q$ is not T or $P \overset{\triangleleft}{\wedge} R$ is U iff either R is U or R is T but Q is not T or R is U or R is T and P is U iff either R is U or R is T and P is U or R is T and Q is not T. **(4)** $(Q\downharpoonleft P) \overset{\diamond}{\wedge} R$ is F iff either $Q\downharpoonleft P$ is F or R is F iff either Q is T and P is F or R is F; $(R \overset{\diamond}{\rightarrow} Q)\downharpoonleft(P \overset{\diamond}{\wedge} R)$ is F iff $R \overset{\diamond}{\rightarrow} Q$ is T and $P \overset{\diamond}{\wedge} R$ is F iff P is F and Q is T or P is F and R is F or R is F iff R is F or Q is T and P is F. ■

Now that projection of presuppositions is formally defined let us see how it helps in finding out the presuppositions of an arbitrary formula. Let us define the **generic presupposition** of P to be such a Q that $P \triangleright R$ iff $Q \vdash_3 R$. Such a generic presupposition always exists; just take $Q = P \overset{|}{\vee} \neg P$; moreover, it is unique up to deductive equivalence. Because if $\widehat{Q}$ is another such generic presupposition then $Q \vdash_3 \widehat{Q}$ as well as $\widehat{Q} \vdash_3 Q$. It does not follow, however, that generic presuppositions are equivalent! By Proposition 4.2, $P \overset{|}{\vee} \neg P, P \overset{\triangleleft}{\vee} \neg P, P \overset{\triangleright}{\vee} \neg P$ and $P \overset{\diamond}{\vee} \neg P$ are all generic presuppositions. These easy solutions have a serious disadvantage. If P contains $\downharpoonleft$, so does $P \overset{|}{\vee} \neg P$, but we like to have a generic presupposition free of $\downharpoonleft$.

Theorem 4.9 *Let $Q_2\downharpoonleft Q_1$ be a presuppositional normal of P. Then $(Q_1 \overset{|}{\vee} \neg Q_1) \overset{|}{\wedge} Q_2$ is a generic presupposition of P and free of $\downharpoonleft$.*

Proof. It is enough to show $P \overset{|}{\vee} \neg P \vdash_3 (Q_1 \overset{|}{\vee} \neg Q_1) \overset{|}{\wedge} Q_2 \vdash_3 P \overset{|}{\vee} \neg P$.

$$\begin{aligned} P \overset{\shortmid}{\vee} \neg P &\equiv_3 Q_2 \downarrow Q_1 \overset{\shortmid}{\vee} \neg(Q_2 \downarrow Q_1) \\ &\equiv_3 Q_2 \downarrow Q_1 \overset{\shortmid}{\vee} Q_2 \downarrow (\neg Q_1) \\ &\equiv_3 (\nabla Q_2 \overset{\shortmid}{\wedge} Q_1) \overset{\shortmid}{\vee} (\nabla Q_2 \overset{\shortmid}{\wedge} \neg Q_1) \\ &\equiv_3 \nabla Q_2 \overset{\shortmid}{\wedge} (Q_1 \overset{\shortmid}{\vee} \neg Q_1) \end{aligned}$$

Together with $\nabla Q_2 \vdash_3 Q_2 \vdash_3 \nabla Q_2$ the claim quickly follows. ■

Now let $\mathbb{L}$ be a language of boolean connectives with locks, and as above $\mathbb{L}^{\downarrow}$ its $\downarrow$-extension. If $P \in \mathbb{L}^{\downarrow}$, denote by $P[Q \downarrow \overline{p}/\overline{p}]$ the result of uniformly substituting $Q \downarrow p$ for p for every variable p of P.

Theorem 4.10 $Q \downarrow P \equiv_3 P[Q \downarrow \overline{p}/\overline{p}]$.

Proof. $Q \downarrow P$ is true iff P and Q are true iff $P[Q \downarrow \overline{p}/\overline{p}]$ is true. For if Q is not true, all $Q \downarrow p$ are undefined and so is $P[Q \downarrow p/\overline{p}]$. But by induction, P is undefined if all variables are U. Hence Q must be true and then $P[Q \downarrow \overline{p}/\overline{p}]$ reduces to P. Thus $Q \downarrow P$ is false iff P is false and Q is true iff $P[Q \downarrow \overline{p}/\overline{p}]$ is false. ■

5 Separation: Internal or External?

At first glance, separation does not look problematic; but a short investigation into the formal prerequisites that make separation possible will produce surprises. To begin we investigate the relationship between two- and three-valued logic. Since in our situation we have defined our three-valued logics as extensions of classical logic it seems straightforward to imitate two-valued logic within three valued logic but more problematic to interpret three-valued logic in two valued logic. But so it only seems.

To begin let us assume that we have some variable p. A priori, p can assume three values, T,F and U. Now define operations p^{Δ} and p^{∇} with the following properties. (i) Both p^{Δ} and p^{∇} are propositions, i. e. have only classical truth values, (ii) $p \equiv_3 p^{\nabla} \downarrow p^{\Delta}$. p^{Δ} and p^{∇} by definition solve the separation problem. It is easy to show, however, that no system using connectives with whatever control structure can produce p^{∇} and p^{Δ}. The reason is simply that any term of that system assumes U if all variables are U. There is no way to define propositions from sentences that are not necessarily propositions themselves. Nevertheless, we might argue that we have considered a system that is too weak; we might, for example, add weak negation.

	$\sim$
T	F
F	T
U	T

Then $p^{\nabla} \equiv_3 \sim(p \overset{|}{\vee} \neg p)$ and $p^{\Delta} \equiv_3 \sim\sim p$. In that case, these two functions become definable. We can on the other hand define weak negation from the assertion function by $\sim p \equiv_3 \neg p^{\Delta}$. Furthermore, $\sim p \equiv_3 p^{\nabla} \overset{\triangleright}{\vee} \neg p$ and so in principle one of the three functions is sufficient to define the others (given enough of the other connectives).

The above arguments show that it is not clear that we can separate any sentence *language internally*. It is, however, always possible to separate sentences *language externally* by stipulating two functions $(-)^{\nabla}$ and $(-)^{\Delta}$ that produce the assertion and proposition of that sentence in another language. If we add certain functions (e. g. weak negations) to our language the external functions can be mimicked internally as we have seen, but the system with external functions has some conceptual advantages for natural language analysis. With the three valued projection algorithm we can formulate principles of $^{\nabla}$- and $^{\Delta}$- percolation. For example,

$$
\begin{array}{lcllcl}
(P \overset{|}{\wedge} Q)^{\nabla} & = & P^{\nabla} \wedge Q^{\nabla} & (P \overset{|}{\wedge} Q)^{\Delta} & = & P^{\Delta} \wedge Q^{\Delta} \\
(P \overset{\triangleright}{\wedge} Q)^{\nabla} & = & P^{\nabla} \wedge (P^{\Delta} \rightarrow Q^{\nabla}) & (P \overset{\triangleright}{\wedge} Q)^{\Delta} & = & P^{\Delta} \wedge Q^{\Delta} \\
(P \overset{\triangleleft}{\wedge} Q)^{\nabla} & = & Q^{\nabla} \wedge (Q^{\Delta} \rightarrow P^{\nabla}) & (P \overset{\triangleleft}{\wedge} Q)^{\Delta} & = & P^{\Delta} \wedge Q^{\Delta} \\
(P \overset{\diamond}{\wedge} Q)^{\nabla} & = & [P^{\nabla} \wedge (P^{\Delta} \rightarrow Q^{\nabla})] \vee [Q^{\nabla} \wedge (Q^{\Delta} \rightarrow P^{\nabla})] & (P \overset{\diamond}{\wedge} Q)^{\Delta} & = & P^{\Delta} \wedge Q^{\Delta}
\end{array}
$$

Notice that the right hand side does not contain the three valued-connectives; this is because we use an external separation in two-valued logic. If we separate internally we can use any of the control equivalents on the right hand side since only the classical values count. Notice also that the assertive part is rather regular in its behaviour.

The discussion on separation becomes less academic when we look at semantics. With few exceptions, semantical theories use two valued logic: Montague Semantics, Discourse Representation Theory, Boolean Semantics, etc. [Sa 90] presents an account of presupposition within DRT. Interestingly, however, he makes no attempt to solve the problem of how to mediate between two- and three-valued logic. Of course, he doesn't see such a problem arise because he views presuppositions as kinds of anaphors so that there is no projection, just allocation. But this means that he closes his eyes in front of some problems. Firstly, there is a marked difference between presuppositions being allocated *as presuppositions* and presuppositions being allocated as *antecedents* or *conjuncts*; we will return to this later. Secondly, the additional conceptual layer of two-valued predicates needs arguing for; it is comfortable to have it but arguably quite unnecessary. Let us consider an example.

(9) Hilary is unmarried.

(9†) $\boldsymbol{unmarried'}(h)$

(9‡) $\mathbf{unmarried'}(h)$

A direct translation of (9) is ($9^\dagger$) which uses a three-valued prediate $\lambda x.\boldsymbol{unmarried}'(x)$ which we simply write as $\boldsymbol{unmarried}$. The two-valued equivalent is $\lambda x.\mathbf{unmarried}'(x)$ or simply **unmarried**. The two are not the same; it is clear that we would not say of a desk, a star or an ant that it is unmarried. Neither would we say this of a six-year old child. In logic we say that *unmarried* is type restricted. It applies only to objects of a certain type. This type restriction is a presupposition since it stable under negation. In the same way *bachelor* is type restricted to all objects that are male and satisfy the type restriction of *unmarried*. There is a whole hierarchy of types which are reflected in the net of elementary presuppositions carried by lexical items. The predicate **unmarried** is not type restricted hence only truth-equivalent to $\boldsymbol{unmarried}$. Indeed, we have exactly $\mathbf{unmarried} \equiv_3 \boldsymbol{unmarried}^{\Delta}$. It is difficult to verbalize this. To say that (9) asserts ($9^\ddagger$) is no proof that such an entity exists; language does not allow to introduce a simple way to express ($9^\ddagger$) without introducing standard presuppositions. This claim on my side needs arguing, of course. Prima facie nothing excludes there being such a predicate conforming to **unmarried**. I would, however, not go as far as that. All I am saying is that there is no straightforward, logical procedure to verbalize the assertional and presuppositional part of a sentence. The operations $(-)^{\Delta}$ and $(-)^{\nabla}$ are merely theoretical devices, and weak negation does not exist contrary to what is sometimes claimed. There is no way of saying simply (9) or (10) to mean ($10^\ddagger$). An argument using (11) as evidence for weak negation begs the question because it is clear that the added conflicting material serves to identify the presupposition that is cancelled. If it is dropped the negation is interpreted as strong. In addition, I still find such examples of questionable acceptability.

(10) Hilary is not married.

($10^\ddagger$) $\neg\mathbf{married}'(h)$

(11) ? Hilary is not married; she is only six years old!

In a similar we can see that spelling out the two-valued presupposition of *unmarried* is not straightforward. In a first attempt we write $\boldsymbol{unmarried}^{\nabla} = \boldsymbol{human} \wedge \boldsymbol{adult}$ but we find that *adult* itself has type restrictions.

6 Dependencies and Allocation

[Sa 90] recently offered a rather detailed account of a theory of allocation. We will not try to improve here on the empirical coverage of that theory; rather we will note some deficiencies of the theory itself. Before we start the analysis, let us quote from that paper. Van der Sandt makes the following basic claim: 'Presuppositions are simply anaphors. They only differ from pronouns or other kinds of semantically less loaded anaphors in that they contain enough descriptive content to establish a reference marker in case the discourse does not provide one.' His formal analysis uses DRT with semantic anchors. Typical entries for presupposition inducers are the following; the anchor is denoted here by a dashed box.

the A

$$\begin{array}{|l|}\hline \quad \alpha_0 \quad \\ \hline A(\alpha_0) \\ \alpha_0 =? \\ \hline \end{array}$$

his/her A

$$\begin{array}{|l|}\hline \quad \alpha_0 \quad \alpha_1 \quad \\ \hline A(\alpha_0) \\ \mathbf{of}(\alpha_0, \alpha_1) \\ \alpha_0 =? \\ \alpha_1 =? \\ \hline \end{array}$$

Firstly, this theory uses classical logic. This might be a surprising diagnosis because it is explicitly states that the Frege-Strawsonian theory of presupposition can be reinstalled by using a truth-value gap. Yet, van der Sandt claims that this gap arises from violating explicit binding constraints and so is in effect reducible to binding. Furthermore, he makes use of separation. We have argued against that earlier; I will take the opportunity to discuss a further disadvantage of internal separation. If presuppositions and assertions of words such as *bachelor* are separable internally and thereby in principle arbitrarily assignable, why does this extra freedom never get exploited? It is namely possible to define a different concept, say, *lachelor*, which has the truth conditions of a bachelor but is presupposes that the object is a living thing. Thus ***lachelor*** $\equiv$ **living** $\downharpoonleft$ **bachelor**; ***lachelor*** is truth-equivalent with ***bachelor*** but evidently not falsity equivalent. Yet such a concept is not lexicalized in the language and it is quite difficult to imagine such a concept. So it seems that presuppositions are just part of the concept and not dissociable from it not the least because we have argued against an additional two-valued interpretation language for reasons of economy. Hence, the story of reallocation (which I also used in the introduction) is just a bad metaphor. The presupposition cannot be freed from the concept; this seems to be at least intuitively accepted (see [He 90]) even though this and other logical implementations do not take notice of that fact. Hence, if presuppositions do not move, they have in some sense to be *copied*. We can understand this as follows. Knowing that at a certain point we have to satisfy certain presuppositions (which are directly given by the concept) we decide to accommodate this presupposition at some place so that the presupposition in question is satisfied at the point we come to evaluate it. It is, however, important (and has been overlooked by van der Sandt) that the reallocated presupposition functions as a presupposition. In the semantics we thus need to postulate the connectives ∇ or $\downharpoonleft$ as primitives in order to guarantee that the inserted material can function as a presupposition not just as an assertion.

The next problem to be considered is the reduction to binding conditions. Probably this can be made meaningful in the following way.

```
program NULL;              program NULL2;
  begin                      N : integer;
    N := 0;                    begin
  end.                           N := 0;
                               end.
```

If we compare these two Pascal programs we see that the left program will fail because the variable N has not been declared. In the other this has been done and so it operates successfully. In the same way we can understand the functioning of the upper part of the box in DRT. It makes a DRS to undefined if a variable is used but not defined in the head section. So in the DRSs below the left DRS is ill-formed while the one to the right is well-formed.

$$\begin{array}{|c|}\hline \\ \hline \mathbf{man}'(x) \\ \hline \end{array} \qquad\qquad \begin{array}{|l|}\hline x \\ \hline \mathbf{man}'(x) \\ \hline \end{array}$$

This would be enough to exclude unbound occurrences if we assume some extra control principles handling composite structures; however, van der Sandt chose to translate lexical items in such a way that the binding conditions are satisfied and he artificially adds the clause $\alpha_0 =?$ to make binding required. I do not see how this can be superior to an account where the variable mechanism itself is exploited. Prior to inserting a DRS for a lexical item we have to choose anyway the variables we are going to insert and we can rely totally on this mechanism to create the presuppositional effect.

Thirdly, it can be demonstrated that presuppositions are as matter of fact not reducible to anaphor resolution. There are obvious arguments against this. One is that there are presuppositions that simply have no variable to be bound. These are easy cases. A trickier example is this one.

(12) If John buys a car he removes the spare wheel.

(13) ? If John buys a car he removes his spare wheel.

(12) could in principle be interpreted in two ways; the presupposition can stay or be reallocated at the top-level. People prefer the first alternative simply because cars tend to come with spare wheels. On the other hand it is possible that in certain contexts (12) presupposes that there is a definite spare wheel e. g. when the previous discourse established a particular spare wheel already. So the choice where the presupposition is accommodated is not governed by binding facts but by some kind of pragmatic *dependency*. Choosing spare wheels to be dependent in some form on cars we opt for a local reading of the presupposition. By default, however, presuppositions or objects are independent. Specific knowledge, in this case about cars and spare wheels, is required to establish such a dependency. We can use the context to create such dependencies or overrun them. (13), however, is the interesting bit of the argument. The theory of [Sa 90] has no means to tell

us why this sentence is so odd. It is not the choice of the locus of the presupposition that is at stake but the question whether the spare wheel is actually identifiable with the one that arises naturally with the car if the presupposition is reallocated at the top. This has nothing to do with binding; we can in principle have it either way.

A theory of allocation must be pragmatic in nature as we have seen. Van der Sandt agrees with that because he limits the locus not only by binding conditions but also by pragmatic factors. But with respect to the last he remains rather vague. He only considers cases in which there appears a logical conflict if the locus of the presupposition is too high. This leaves the impression that logic plays a significant role as a determining factor. I am tempted to say that it does not. It seems to me that this theory should be based on a theory of *dependencies*. I used this term earlier but let me be a bit more precise here what I mean by that. Dependencies are relations that hold between objects, between concepts or between objects and concepts etc. In logic, when we use arbitrary objects (see [Fi 85]) an object that is freed from an existential quantifier depends on all the free variables or constants of the formula. So, $(\forall x)(\exists y)\phi(x,y)$ does not imply $(\exists y)(\forall x)\phi(x,y)$ because if we were to free objects from the quantifiers we get dependencies that block the reintroduction of the quantifier for an object. This goes as follows. We first assume an a such that $(\exists y)\phi(a,y)$. Then we take a b such that $\phi(a,b)$. The conclusion that the conclusion $(\forall x)\phi(x,b)$ is not licenced because b depends on a. In this way the theory by Fine shows how reasoning with quantifiers is reducible to reasoning with objects. The dependencies are the main thing that we have to memorize if we want to reason correctly. Notice that if we manage to prove that b actually does not depend on the choice of a the above reasoning could be carried on and the implication $(\forall x)(\exists y)\phi(x,y) \to (\exists y)(\forall x)\phi(x,y)$ would then hold. The observable is therefore the dependency of b on a. How we recognize this dependency is quite another story but it certainly determines the reasoning. In totally the same way we understand the mechanism of reallocation of presuppositions. *Dependency* is a primitive notion; by some means we come to recognize that the truth of some proposition or the referent of some description is dependent on some other. It is then a consequence that the dependent description or proposition cannot be processed before the one on which it depends and this in turn explains why the locus of a presupposition must be inside the scope of all things on which the presupposition depends. A final illustration is this sentence.

(14) As long as the moon wanders around the earth fishermen will love the tide.

It is known (not to all of us) that the moon creates the tide. So, *the tide* ceases to refer as soon as moon stops wandering. Hence the locus of the presupposition is its origin. But those who fail to know (or notice) that will read this as saying that there is a tide whether or not the moon is wandering around the world. The two differ not only logically; in the latter reading we are led to think that it is the moon which induces the love of the fishermen for the tide in some way. The latter therefore leads us to see a dependency between the moon's wandering around the earth and fishermen's love of the tide whereas in the first case no such dependency is induced.

7 Conclusion

In comparing possible extensions of two-valued logic to three-valued logic using explicit control structures we have managed to give an account of standard presuppositional behaviour of computers or mathematicians. We have boosted this up to natural language semantics by assuming possible reallocation of presupposition. The latter extension touches on pragmatics and is therefore not easily spelt out in detail. We will in this last section evaluate the pros and cons not of the theory of allocation but of the interpretation in three-valued logic.

It has been noted that standard three-valued logics for presupposition can be rephrased with the help of the rules of *context change*. Underlying that is the notion of presupposition as failure. This would lead to Bochvar's Logic was it not the case that the evaluation procedure is spelt out differently. Let us study the following two clauses (cf. [He 90]).

(lc$\rightarrow$) If the local context for *if* A *then* B is X, the local context for A is X and the local context for B is $X \cup \{A\}$.

(lc$\vee$) If the local context for A *or* B is X then the context for A is X and the local context for B is $X \cup \{\neg A\}$.

Each of A and B may carry their own presuppositions but in (**lc** $\rightarrow$) the presuppositions for B are evaluated only in those situations where not only X holds but also A. Taking this together with the standard two-valued interpretation of $\rightarrow$ yields the truth-tables for $\overset{\triangleright}{\rightarrow}$. This is not hard to check. Similarly, (**lc**$\vee$) leads to $\overset{\triangleright}{\vee}$. The problems that arose were that (1) the local context is not fixed by the connective and (2) one cannot freely assign any rule of local context to a connective. We cannot, for example, choose to take $X \cup \{\neg A\}$ as the context for B in (**lc** $\rightarrow$). [He 90] is particularly worried by this. But the problem is that too much is specified in the rules of local context. We have seen earlier that the control structure is enough; so rather than anticipating the actual context against which B is evaluated we only say that A has to be evaluated first and B is evaluated against the context that results from X by adding the condition that must be satisfied if the computer is about to process B. This readily explains the difference between the rules of local context of (**lc** $\rightarrow$) and (**lc**$\vee$). We have seen that the control structures are independent of the connective and that the connective plus the control structure yield a definite truth-table. Each of the possible combinations is realized in language. As examples we study (15) and (16). If we consider all four possible options we see that (15) and (16) are free of presupposition if the context rules are spelled out as (**lc**$'\vee$) and (**lc**$'$ $\rightarrow$). They sound rather circular but in fact reflect the control strategy of $\overset{\diamond}{\vee}$ and $\overset{\diamond}{\rightarrow}$.

(15) Either John has started smoking or he has just stopped smoking.

(16) If John hasn't started smoking he has just stopped smoking.

(**lc**$'\vee$) If X is the local context for *either* A *or* B then the local context for A is $X \cup \{\neg B\}$ and the local context for B is $X \cup \{\neg A\}$.

(**lc**$' \rightarrow$) If X is the local context for *if* A *then* B then the local context for A is $X \cup \{\neg B\}$ and the local context for B is $X \cup \{A\}$.

We can perform the same trick with implication and thereby force a symmetrical reading or the implication. It remains to be seen, however, what exactly determines this choice of the control structure. This we have not been able to establish nor the conditions under which it may take place at all; nor how this relates with the dependencies.

References

[Bl 86] Blamey, Steven: *Partial Logic*, in: Gabbay & Guenthner (eds.): Handbook of Philosophical Logic, vol. 3, Reidel, Dordrecht, 1986, 1 - 70

[Bo 81] Bochvar, D. A.: *On a Three-valued Logical Calculus and its Applications to the Analysis of the Paradoxes of the Classical Extended Functional Calculus*, translated by Merrie Bergman, History and Philosophy of Logic 2(1981), 87 - 112

[Fi 85] Fine, Kit: *Natural Deduction and Arbitrary Objects*, Journal of Philosophical Logic 14(1985), 57 - 107

[Fr 68] van Fraassen, B.: *Presupposition, Implication and Self-Reference*, Journal of Philosophy 65(1968), 135 - 152

[Ha 75] Hayes, P.: *Three-valued Logic and Computer Science, Part I: Propositional Calculus and 3-valued inference*, Ms. of the University of Essex, 1975

[He 90] Heim, I.: *Presupposition Projection*, Workshop on Presupposition, Lexical Meaning and Discourse Processes, 1990

[Ka 79] Karttunen, L.: *Presupposition and linguistic context*, Theoretical Linguistics 1(1979), 181 - 194

[KP 79] Karttunen, L. and Peters, S.: *Conventional implicature*, in: Oh and Dinneen (eds.): *Synatx and Semantics 11: Presupposition*, Acacdemic Press, New York, 1979, 1 - 56

[Ke 75] Kempson, R. M.: *Presupposition and the delimitation of semantics*, Cambridge University Press, Cambridge, 1975

[La 71] Langendoen, D. H., Savin, L.: *The projection problem for presuppositions*, in: Fillmore & Langendoen: *Studies in Linguistic Semantics*, Holt, New York, 1971, 5.2 - 6.2

[La 89] Langholm, Thore: *Partiality, Truth and Persistence*, CSLI Lecture Notes No. 15, 1989

[Ra 79] Rautenberg, W.: *Klassische und nichtklassische Aussagenlogik*, Vieweg Verlag, Wiesbaden, 1979

[Sa 88] van der Sandt, Rob A.: *Presupposition and Context*, Croom Helm Linguistic Series, London, 1988

[Sa 90] van der Sandt, R. A.: *Ananphora and accommodation*, Workshop on Presupposition, Lexical Meaning and Discourse Processes, 1990

[Ur 86] Urquhart, Alasdair: *Many-valued logic*, in: Gabbay & Guenthner (eds.): Handbook of Philosophical Logic, vol. 3, Reidel, Dordrecht, 1986

[Vi 92] Visser, Albert: *Actions under Presuppositions*, Preprint No. 76, Logic Group of the Dept. of Philosophy, University of Utrecht, 1992

[Wó 88] Wójcicki, A.: *Theory of logical calculi*, Kluwer, Dordrecht, 1988