Compositional multidimensionality and the lexicon-semantics interface

Daniel Gutzmann

University of Frankfurt, 60629 Frankfurt, Germany,
gutzmann@lingua.uni-frankfurt.de, home page: http://www.gutzmann.org

Abstract. We provide a reformulation of the multidimensional semantics of the Pottsean $\mathcal{L}_{\mathrm{CI}}$ and its extensions which (i) solves issues of compositionality and (ii) reduces the amount of composition rules. This is done by changing to a system that embraces true compositional multidimensionality, so that every expression is represented by three fully specified for meaning dimensions in the logic. Restrictions on possible expressions and combinations are implemented at the lexicon-syntax interface of the logical language that extend otherwise at most 2-dimensional expressions into 3-dimensional objects that can be used in the semantic derivations. This helps to keep the lexical entries simple and allow to account for possible cross-linguistic variance without needing to change the underlying logic.. . .

Keywords: multidimensional semantics, use-conditions, expressives, compositionality, type-driven translation

1 Introduction

Thanks to the influential book by [1], conventional implicatures and expressive meaning have received a lot of well-deserved interest during the last years and the multidimensional framework that are offered by that work have been fruitfully applied to various phenomena. However, the original logic $\mathcal{L}_{\mathrm{CI}}$ has been argued to be too restrictive, and therefore, several extensions have been developed to broaden the empirical scope of that approach [3, 2]. However, as far as these frameworks – which I collectively refer to as $\mathcal{L}^*_{\mathrm{CI}}$ – widened our understanding of how expressive meaning interacts with other meaning components, there are two issues that I will address in this talk. The first one regards the general approach to multidimensional meaning taken in the mentioned works and is connected with worries about compositionality. The second regards the proliferation of types and combinatoric rules in the extensions. The third one regards specific constructions (abstraction, quantification) that cannot be analyzed due to the way the composition works in $\mathcal{L}^*_{\mathrm{CI}}$. As I will argue, all three issues can be solved by the same modification of the framework.

Before going on, a brief terminological remark. Instead of using the term *expressive* meaning as introduced by [4], I rather prefer to call the meaning contribute by the phenomena under discussion *use-conditional* [5], for the simple

reason that the class of expression that contribute the kind of projective meaning modeled by $\mathcal{L}^*_{\mathrm{CI}}$ goes beyond what stereotypically is conceived as expressives. That is, beside standard examples of expressive adjectives, honorifics, or ethnical slurs, there are also expressions like particles in German [5] or Japanese [6] or even syntactic constructions [7] that fall under the scope of the framework provided by the formal systems I will discuss in this paper. Accordingly, I will speak of expressions that contribute use-conditional meaning as *use-conditional items* or UCIs for short.

2 $\mathcal{L}_{\mathrm{CI}}$ and its extensions

In the following, I will give a brief outline of $\mathcal{L}_{\mathrm{CI}}$ and its extensions to illustrate how they approach the core data.

The data that the original $\mathcal{L}_{\mathrm{CI}}$ [1] has been developed for is constituted by what I call expletive (functional) UCIs [8]. By this, I mean expressions that, once applied to their truth-conditional argument, express just use-conditional content (UC). That is, adding them to or removing them from a sentence does not alter its truth-condition content (TC). Standard examples are expressive adjectives.

(1) That damn Kaplan got promoted.(5)
TC: Kaplan got promoted.
UC: The speaker has a negative attitude towards Kaplan.

From a formal perspective such UCIs are distinguished from ordinary truth-conditional items by having a semantic type that involves a use-conditional type in its output. $\mathcal{L}_{\mathrm{CI}}$ therefore extends the common type definition in (2) by a new basic expressive/use-conditional types and a corresponding recursive definition for complex use-conditional types as in (3).

(2) *Ordinary truth-conditional types*
a. e, t, s are basic truth-conditional types.
b. If σ, τ are truth-conditional types, $\langle \sigma, \tau \rangle$ is a truth-conditional type.

(3) *Simple use-conditional*
a. u is a basic use-conditional type. ("use-conditional proposition")
b. If σ is a truth-conditional and τ is a use-conditional type, $\langle \sigma, \tau \rangle$ is a use-conditional type.

The combinatorics of those new types is regulated by a corresponding new composition rule for use-conditional application rule.[1]

(4) $$\frac{\alpha : \langle \sigma^a, \tau^c \rangle \qquad \beta : \sigma^a}{\beta : \sigma^a \bullet \alpha(\beta) : \tau^c}$$

[1] Instead of the tree notation used in [1], I will use the proof-style notational variant employed in [2].

This rule ensure that if a UCI combines with its argument, it is isolated from the truth-conditional content (indicated by the $\bullet$). Its argument is passed along unmodified. For this reason, use-conditional application has been thought of as being "non-resource sensitive".

In addition to this rule, we also need an elimination rule that strips off saturated use-conditional content so that it does not interfere with the truth-conditional content for the rest of the derivation.

(5) $$\frac{\beta : \tau^a \bullet \alpha : t^c}{\beta : \tau^a}$$

These two new rule, together with ordinary functional application, can derive examples like .

(6) That damn Kaplan got promoted.

$$\dfrac{\dfrac{\dfrac{\textbf{damn} : \langle e, u\rangle \qquad \textbf{kaplan} : e}{\textbf{kaplan} : e \bullet \textbf{damn}(\textbf{kaplan}) : u}}{\textbf{kaplan}} \qquad \textbf{got-promoted} : \langle e, t\rangle}{\textbf{go-promoted}(\textbf{kaplan}) : t}$$

The problem with this set of types and rules, offered by $\mathcal{L}_{\text{CI}}$ is, however, that it has been shown to be too restrictive, as it can only deal with such expletive UCI that do not interact with a sentence's truth-conditional content. For specifically, it can neither deal with mixed expressives [2, 3] and so-called shunting UCIs that lead to pure use-conditional content [2], nor with use-conditional modification [3]. For instance, *Kraut* in (7) contributes truth-conditional and use-conditional content simultaneously. Its predicated being German in the truth-conditional dimension, while expressing a negative attitude towards Germans in the use-conditional tier.

(7) Lessing was a Kraut.
TC: Lessing was a German.
UC The speaker has a negative attitude towards Germans.

Shunting UCIs, on the other hand, do neither contribute truth-conditional content like mixed UCIs nor are they non-resource-sensitive as expletive UCIs are. They do not pass back their argument but take it them, so to speak, without leaving anything back in the truth-conditional dimension. An example is the exclamative operator that is arguably present in exclamatives like (8) and which leads to a speech act that does only have use-conditional content [9].

(8) How tall he is!
TC: –
UC: The speaker is surprised by his degree of tallness.

Finally, there are expression that modify other use-conditional items as in (9), which is also not accounted for by the original $\mathcal{L}_{\mathrm{CI}}$.

(9) That fucking bastard Kaplan got promoted.
TC: Kaplan got promoted.
UC: The speaker has a highly negative attitude towards Kaplan.

To account for those all those cases, $\mathcal{L}_{\mathrm{CI}}$ has been extended by additional types and compositions rule. That is, besides the ordinary truth-conditional types in (2) and (3), we now have new basic and complex types for shunting UCIs as in (10), as well as new recursive definitions for mixed (11) and pure use-conditional types (12).

(10) *Shunting types*
a. u^s is a shunting type.
b. If σ a is truth-conditional or shunting type, and τ is a shunting type, $\langle \sigma, \tau \rangle$ is a shunting type.

(11) *Mixed types*
If σ, τ are truth-conditional types and ρ is a shunting type, then $\langle \sigma, \tau \rangle \times \langle \sigma, \rho \rangle$ is a mixed type.

(12) *Pure use-conditional types*
If σ, τ are simple expressive types then $\langle \sigma, \tau \rangle$ is an expressive type.

These new types of course need corresponding composition rules, which are also added to the inventor of $\mathcal{L}^*_{\mathrm{CI}}$: we now have rules for shunting, mixed, and pure application as well as rule for mixed elimination. That is, the full range of types for $\mathcal{L}^*_{\mathrm{CI}}$ is given by the definition in (2), (3) and (10)–(12), while the full set of composition rules is as given in Fig. (12).

Even if the extended set of types and composition rules of $\mathcal{L}^*_{\mathrm{CI}}$ is able to overcome the restrictiveness of the original $\mathcal{L}_{\mathrm{CI}}$ and thereby leads to a better coverage of the empirical data, it comes with a conceptual cost, as lot of the initial appeal of $\mathcal{L}_{\mathrm{CI}}$– its relatively simplicity in terms of the combinatorics and type extensions – gets lost. However, we will show that the amount of types and rules can be reduced by the same strategy that solves the compositionality issue.

3 Compositionality

As we have seen in (6), the basic idea of how the composition of the two meaning dimensions in $\mathcal{L}^*_{\mathrm{CI}}$ works can be sketched as follows. If a use-conditional item reaches propositional status, it becomes isolated from the descriptive content (indicated by the bullet $\bullet$) and then is "stranded" inside the derivation, such that it is inaccessible for further modification.

So far, this is however, only one part of the story. Of course, we somehow want the use-conditional content dangling inside the derivation to be interpreted after all. This is achieved by a mechanism, called parse tree interpretation [1]. What

Functional application

$$\frac{\alpha : \langle \sigma^a, \tau^a \rangle \qquad \beta : \sigma^a}{\alpha(\beta) : \tau^a}$$

Expressive application

$$\frac{\alpha : \langle \sigma^a, \tau^c \rangle \qquad \beta : \sigma^a}{\beta : \sigma^a \bullet \alpha(\beta) : \tau^c}$$

Shunting application

$$\frac{\alpha : \left\langle \sigma^{\{a,s\}}, \tau^s \right\rangle \qquad \beta : \sigma^{\{a,s\}}}{\alpha(\beta) : \tau^s}$$

Mixed application

$$\frac{\alpha \blacklozenge \beta : \langle \sigma^a, \tau^a \rangle \times \langle \sigma^a, \upsilon^s \rangle \qquad \gamma : \sigma^a}{\alpha(\gamma) \blacklozenge \beta(\gamma) : \tau^a \times \upsilon^s}$$

Pure application

$$\frac{\alpha : \left\langle \sigma^{\{s,c\}}, \tau^c \right\rangle \qquad \beta : \sigma^{\{s,c\}}}{\alpha(\beta) : \tau^c}$$

Expressive elimination

$$\frac{\beta : \tau^a \bullet \alpha : t^c}{\beta : \tau^a}$$

Mixed elimination

$$\frac{\alpha \blacklozenge \beta : \sigma^a \times t^s}{\alpha : \sigma^a \bullet \beta : t^c}$$

Fig. 1. Composition rules for $\mathcal{L}^*_{\mathrm{CI}}$

is does is that instead of merely interpretation the root of a derivation – which corresponds to the truth-conditional content except if shunting UCIs removed it – it interprets the entire proof, so that so that one arrives at an interpreted pair whose first projection is the sentence's truth-conditional content (i.e. the root of the proof) and whose second dimension is the collection of all dangling expressive propositions.

(13) *Parsetree interpretation*

The interpretation of a proof tree $\mathcal{T}$ with a term $\alpha : \sigma$ on its root node, and distinct terms $\beta_1 : u, \ldots, \beta_n : u$ on nodes in it is

a. $[\![\mathcal{T}]\!] = \langle [\![\alpha]\!], \{[\![\beta_1]\!], \ldots, [\![\beta_n]\!]\}\rangle$, if σ is a truth-conditional type.

b. $[\![\mathcal{T}]\!] = \langle T, \{[\![\alpha]\!], [\![\beta_1]\!], \ldots, [\![\beta_n]\!]\}\rangle$, if σ is a shunting type, where T is a trivial proposition.

Hence, for example (6), parsetree interpretation delivers us the following interpretation:

(14) $[\![(14)]\!] = \langle [\![\textbf{got-promoted}(\textbf{kaplan})]\!], [\![\textbf{damn}(\textbf{kaplan})]\!]\rangle$

However, there is a problem with this procedure. As it has been noted, this "parsetree interpretation" does not fulfill the ordinary principle of compositionality, according to which only the immediate parts of a complex expression (and

the way in which they are combined) are taken into account in order to calculate the meaning of complex expressions.

> Though formally precise, this method is not compositional. The reason is that the computation of the side-issue content draws information from deeply embedded expressions (the supplement phrases), rather than only from the denotation of the sentences immediate subconstituents. [**?**]

There are some recent proposal to account for expressive content in a way that respects compositionality ([10] use continuations; [11] use monads), but only at the cost of introducing a mightier machinery and using a completely different system. Instead of a complete redesign, we will present a framework that is closer to the spirit of $\mathcal{L}^*_{\mathrm{CI}}$ and is rather a compositional and, we believe, simpler reformulation of the core ideas of $\mathcal{L}^*_{\mathrm{CI}}$.

4 Quantification problems

The third problematic case that we want to discuss is constituted by constructions involving cross-dimensional quantification and which not even the extended type- and composition system of $\mathcal{L}^*_{\mathrm{CI}}$ can deal with.

Due to the way the composition works in $\mathcal{L}^*_{\mathrm{CI}}$, it cannot deal with simple cases of cross-dimensional quantification and/or abstraction. We begin with an example of the former. For the sake of illustration, suppose that *to goggle* is a mixed UCI meaning 'to look at' in the descriptive dimension, while expressing a negative attitude towards the looking in the expressive dimension (represented simply by **bad**). Now, the problem is if, after having combined with its direct object, which is correctly distributed into the two dimensions of the mixed predicate, the resulting expression shall combined with a quantifier in subject position.

(15) Everybody goggles at Klaus.

$$\dfrac{\mathbf{everybody} : \langle\langle e,t\rangle,t\rangle \qquad \dfrac{\mathbf{look\text{-}at} \diamond \lambda x\lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(x)(y)) : \langle\langle e,t\rangle,t\rangle \times \langle e,\langle e,u\rangle\rangle \qquad \mathbf{klaus} : e}{\mathbf{look\text{-}at}(\mathbf{klaus}) \diamond \lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(\mathbf{klaus})(y)) : \langle e,t\rangle \times \langle e,u\rangle}}{\lightning} \;\text{✗}$$

In $\mathcal{L}^*_{\mathrm{CI}}$, the 1-dimensional mixed expressions can only apply to a single argument, but there is no rule that allows a 1-dimensional (1D) expressions (like the quantifier) to apply to a 2-dimensional argument.

Another, more complex case, involves a quantified DP in object position.

(16) Klaus googles at every girl.

The problem of quantifiers in object position is that, under standard typing and surface constituency, the expressions cannot be combined, as to google at is of type $\langle e,\langle e,t\rangle\rangle$ and therefore needs a type e direct object, while *every girl*

is a quantifier of type $\langle\langle e,t\rangle,t\rangle$. A common solution to this mismatch is the assumption of quantifier raising (QR) at LF [12]. The direct object moves to take scope over the entire sentence, leaving an index trace behind, which is bound by an index that is adjoined to the sentence at the position below the one to which the quantifier has been raised. The gives us the following LF structure for (16).

(17) [$[_{\text{QP}}$ every girl$]_i$ [1 [Klaus $[_{\text{VP}}$ googles at t_1]]]]

When this LF is interpreted by the semantics, the trace is interpreted as variable. Crucially, the binding index above the sentence has to be understood as a lambda abstractor binding that variable. However, when we now substitute the semantic representations for the expressions in (17) and compose the complex expressions in accordance with the proof rules of $\mathcal{L}^*_{\text{CI}}$, we arrive at the following derivation.

(18)

$$\dfrac{\mathbf{every}(\mathbf{girl}) : \langle\langle e,t\rangle,t\rangle \qquad \dfrac{\lambda z \quad \dfrac{\dfrac{\dfrac{\mathbf{klaus} : e \qquad \dfrac{\dfrac{\lambda x\lambda y.\mathbf{look\text{-}at}(x)(y) \blacklozenge \lambda x\lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(x)(y)) : \langle e,\langle e,t\rangle\rangle \times \langle e,\langle e,u^s\rangle\rangle}{\lambda y.\mathbf{look\text{-}at}(z)(y) \blacklozenge \lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(z)(y)) : \langle e,t\rangle \times \langle e,u^s\rangle} \quad z : e}{\mathbf{look\text{-}at}(z)(\mathbf{klaus}) \blacklozenge \mathbf{bad}(\mathbf{look\text{-}at}(z)(\mathbf{klaus})) : t \times u^s}}{\mathbf{look\text{-}at}(z)(\mathbf{klaus}) : t \bullet \mathbf{bad}(\mathbf{look\text{-}at}(z)(\mathbf{klaus})) : u^s}}{\mathbf{look\text{-}at}(z)(\mathbf{klaus}) : t}}{\lambda z\mathbf{look\text{-}at}(z)(\mathbf{klaus}) : \langle e,t\rangle}}{\mathbf{every}(\mathbf{girl})(\lambda z\mathbf{look\text{-}at}(z)(\mathbf{klaus})) : t}$$

The problem is that while for the at-issue part of *to google*, the combination of a provisional introduction of the object argument and its later abstraction works as needed, the variable introduced by the trace remains unbound by the lambda operator in the UC dimension, because it is isolated by use-conditional application. This predicts that, in the UC dimension, (17) expresses a negative attitude regarding Heinz's looking at $g(z)$, i.e., whatever referent is assigned to the variable z by the assignment function g. This is of course not the use-conditional content expressed by (17).

5 Compositional multidimensionality

The three mentioned issues can be addressed if we reformulate $\mathcal{L}^*_{\text{CI}}$ in a truely multidimensional way. First note that $\mathcal{L}^*_{\text{CI}}$ is exhibit *interpretational multidimensionality*, as one may call it: Except for mixed-type expressions, the expressions of $\mathcal{L}^*_{\text{CI}}$ all have just one dimension and the multidimensionality is introduced by parsetree interpretation, which distributes expressions in a derivation into the two meaning dimensions. This contrast with *compositional multidimensionality*, which serves as the key to reformulate the core ideas of $\mathcal{L}^*_{\text{CI}}$ in a way that can addressed the raised challenges. The basic idea of compositional multidimensionality is that every natural language expression can systematically be associated with all meaning dimensions. Therefore, during the composition, all dimensions

are calculated at each step based on the dimensions of its daughters. As we will see, this enables a reduction of the type system and combinatoric rules. I call the resulting system $\mathcal{L}_{\mathrm{TU}}$.

The type definition for $\mathcal{L}_{\mathrm{TU}}$ is rather simple, as it only distinguish between truth- and use-conditional types and does not divide the later into further subcategories.

(19) *Types for $\mathcal{L}_{TU}$*

a. e, t are basic truth-conditional types for $\mathcal{L}_{\mathrm{TU}}$.
b. u is a basic use-conditional type for $\mathcal{L}_{\mathrm{TU}}$.
c. If τ is a truth-conditional type for $\mathcal{L}_{\mathrm{TU}}$, then $\langle s, \tau \rangle$ is a truth-conditional type for $\mathcal{L}_{\mathrm{TU}}$.
d. If σ and τ are truth-conditional types for $\mathcal{L}_{\mathrm{TU}}$, then $\langle \sigma, \tau \rangle$ is a truth-conditional type for $\mathcal{L}_{\mathrm{TU}}$.
e. If σ is a type for $\mathcal{L}_{\mathrm{TU}}$ and τ is a use-conditional type for $\mathcal{L}_{\mathrm{TU}}$, then $\langle \sigma, \tau \rangle$ is a use-conditional type for $\mathcal{L}_{\mathrm{TU}}$.
f. The set of all types for $\mathcal{L}_{\mathrm{TU}}$ is the union of all truth-conditional and use-conditional types.

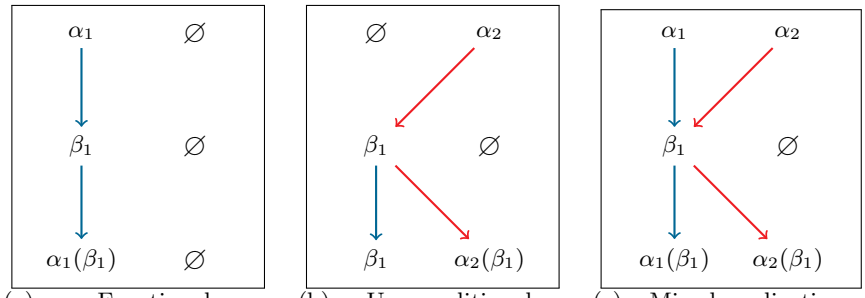
Officially, the expressions in the compositional multidimensional system are triples, but we will use the $\mathcal{L}^*_{\mathrm{CI}}$ separators and write $\langle A, B, C \rangle$ as $A \blacklozenge B \bullet C$. The first dimension is the plain descriptive content (i.e., nothing with type u in it), while the third dimension is used to isolate satisfied expressive content (i.e., expressions of type u), where it can only be merged with other expressive propositions. The key component is the second dimension which functions like some kind of logging system that keeps track of expressions that are "active" for calculating expressive content. I call these dimensions the t-, s-, and u-dimension respectively. (The second and third dimension roughly correspond to what is behind the diamond and the bullet in $\mathcal{L}^*_{\mathrm{CI}}$.)

To see how such a consequently three-dimensional approach can reduce the number of composition rules, let us first recast the rules of $\mathcal{L}^*_{\mathrm{CI}}$ in a three-dimensional way.

We start by focusing on the first two dimensions. In Fig. 2, I employ arrow diagrams for better illustration the flow of information between the t- and s-dimension.

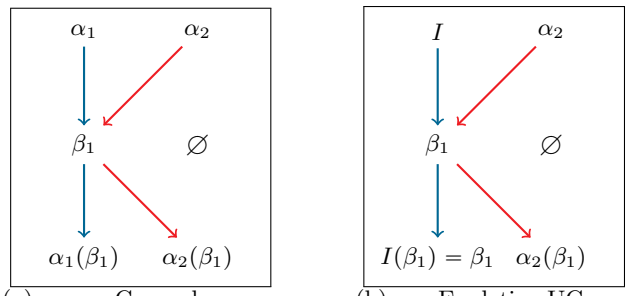
As these diagrams make clear, under such a consequently multidimensional view, simple (=expletive) simple (=expletive) use-conditional application can be viewed as a special instance of mixed application, namely if α_1 is an identity function. This is illustrated in 3.

That is, we have reduced three rules to two. However, I think we can do even better. The key to achieve this lies in the second dimension of the argument expression. Looking at the schematic visualization of functional application in Fig. 2(a) and general use-conditional application in Fig. 3(a), we see that the



(a) Functional application (b) Use-conditional application (c) Mixed application

Fig. 2. Composition of the first and second meaning dimensions in 3-dimensional $\mathcal{L}^*_{\mathrm{CI}}$



(a) General use-conditional application (b) Expletive UC application

Fig. 3. Expletive use-conditional application as an instance of general use-conditional application

argument's s-dimension does not play a role in any of the two applications.[2] Furthermore, it is always the t-dimension of the argument to which both dimensions of the functional expression apply. Now let me employ the following trick. Instead of using the empty set for representing "empty" use-conditional content in the s-dimension of an expression, I instead use a copy of the first dimension. I do this for every empty s-dimension. In case of ordinary functional application, the entire application is therefore replicated in the second dimension. For the moment, I leave the arrows untouched. For illustration, I put the copied material in gray boxes. We thus end up with the two schemata in Fig. 4.

Note that merely copying the truth-conditional content to the s-dimension does not affect the composition in any meaningful way, because the places to

[2] That is not to say that it is irrelevant for the application rules. Quite the contrary. It constraints the use of the application to just those cases in which the s-dimension of the argument is empty.

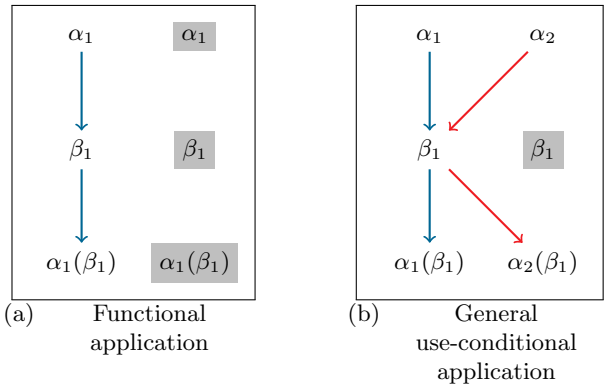


Fig. 4. Composition of the first two meaning dimensions with non-empty dimensions

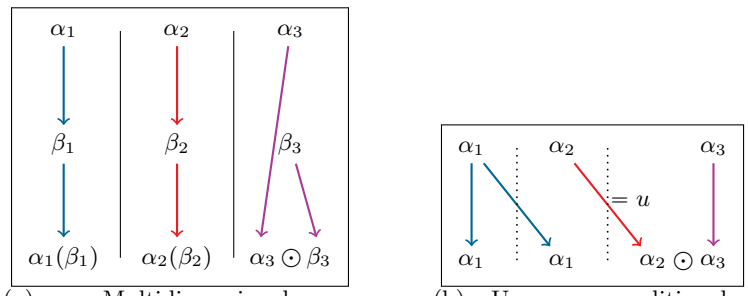
which I have copied material do not play any role in the application schemata. No gray box is connected to anything else.[3]

Comparing the two schemata resulting from the copy-trick reveals how this enables the unification of the two rules. What happens in the t-dimension is the same as before the copying trick. Now, in the s-dimension, we see that the sole difference is that the function's s-dimension may differ from the t-dimension in case of expressive application, while it has to be the same in functional application. This also transfers to the outcome of the application. Now, if the function's s-dimension in expressive application happens to be the same as its t-dimension (i.e $\alpha_2 = \alpha_1$), then the schema for expressive application reads the same as the one for ordinary functional application. That is, functional application can be understood as a special case of general expressive application, namely one in which the function's s-dimension is a copy of its t-dimension. This maneuver then opens up an additional possibility for simplification. Note that if we were to employ general expressive application as in Fig. 4(b) as the most general rule, the s-dimension of the argument would still remain unused. However, since it happens to be a copy of the t-dimension, we can equally assume that the function's s-dimension applies to the s-dimension of the argument, instead of the t-dimension. That is, we can have a entirely *intra*-dimensional application, instead of the *trans*-dimensional application that so far has been the hallmark of the second dimension, since $\mathcal{L}_{\mathrm{CI}}$. Of course, this is currently nothing more then an aesthetic advantage. However, it also allows us to subsume also use-conditional modification by lacing the requirement that the s-dimension of the argument is a copy of its t-dimension.

The new visual illustration for the resulting single, generalized rule for what can be called multidimensional application is given in Fig. 5(a). For complete-

[3] Again, even if the content in the grey boxes does not actively take part in the functional applications inside the entire application rule, it constraints the use of the schema to instance in which the s-dimension is as given by the boxes.

ness, I have also added the composition of the third dimension, which has been put aside during the present reformulation. In addition to the application rule, we also need a new multidimensional elimination rules, which empties the s-dimension by copying saturated use-conditional content to the third dimension (where it merge with other use-conditional propositions by means of the use-conditional conjunction $\odot$) and copying the t- to the s-dimension.



(a) Multidimensional application

(b) Unary use-conditional elimination

Fig. 5. Composition with multidimensional application and unary elimination

Leaving the arrow diagrams, these rules can be stated in the proof-style notation as follows.

(20) $$\frac{\alpha_1 : \langle \sigma, \tau \rangle \blacklozenge \alpha_2 : \langle \rho, \nu \rangle \bullet \alpha_3 : u \qquad \beta_1 : \sigma \blacklozenge \beta_2 : \rho \bullet \beta_3 : u}{\alpha_1(\beta_1) : \tau \blacklozenge \alpha_2(\beta_2) : \nu \bullet \alpha_3 \odot \beta_3 : u}$$

(21) $$\frac{\alpha_1 : \sigma \blacklozenge \alpha_2 : u \bullet \alpha_3 : u}{\alpha_1 : \sigma \blacklozenge \alpha_1 : \sigma \bullet \alpha_3 \odot \alpha_2 : u}$$

6 Lexical extensions

If every lexical expression would correspond to a 3D expression, the lexicon would contain a lot of redundant information, since in most cases, missing dimensions can be deduced once we know one or two dimensions. However, if we want to keep the lexicon simple we need what we call *lexical extension rules* (LERs) that expand the lexical entries into proper 3D expressions that can be used in semantic derivations. This interface between lexical and derivational semantics allows us also to impose desired restrictions, because, as stated so far, the system is quite liberal. In addition, the use of LERs allows us to account for cross-linguistic variation without changing the compositional system of the logic. If a language does not exhibit, say, mixed expressives or expressives modifiers, its lexicon does just not possess the relevant LERs, so that such expressions could never enter the semantic composition. A subset of LERs for languages that do

allow mixed expressives as well as modification of expressive predicates (but not of expressive propositions) is given in (22). T and U range over descriptive and expressive types respectively. $I_\sigma = \lambda x_\sigma . x_\sigma$ is an identity function on expressions of type σ. The expressions T and U are dummy expression for trivial descriptive and expressive content and both denote the set of all possible worlds. Therefore, $[\![\alpha : u \odot U]\!] = [\![\alpha : u]\!]$. The rule in (22)e uses the following convention: for every type α and each $n \geqslant 0$, $\alpha^n = \alpha$ if $n = 0$, and $\alpha^n = \langle \alpha^{n-1}, \alpha^{n-1} \rangle$ if $n > 0$.

(22) *Lexical extension rules*

a. *descriptive expressions:*
$A : \mathrm{T} \Rightarrow A \blacklozenge A \bullet U$

b. *expletive expressives:*
$A : \langle \mathrm{T}, \mathrm{U} \rangle \Rightarrow I_{\mathrm{T}} \blacklozenge A : \langle \mathrm{T}, \mathrm{U} \rangle \bullet U$

c. *mixed expressives:*
$\langle A : \langle \mathrm{T}_1, \mathrm{T}_2 \rangle, B : \langle T_1, U \rangle \rangle \Rightarrow A : \langle \mathrm{T}_1, \mathrm{T}_2 \rangle \blacklozenge B : \langle T_1, U \rangle \bullet U$

d. *shunting expressives:*
$\langle \lambda x_{\mathrm{T}}.T, A : \langle \mathrm{T}, \mathrm{U} \rangle \rangle \Rightarrow \lambda x_{\mathrm{T}}.T \blacklozenge A : \langle \mathrm{T}, \mathrm{U} \rangle \bullet U$

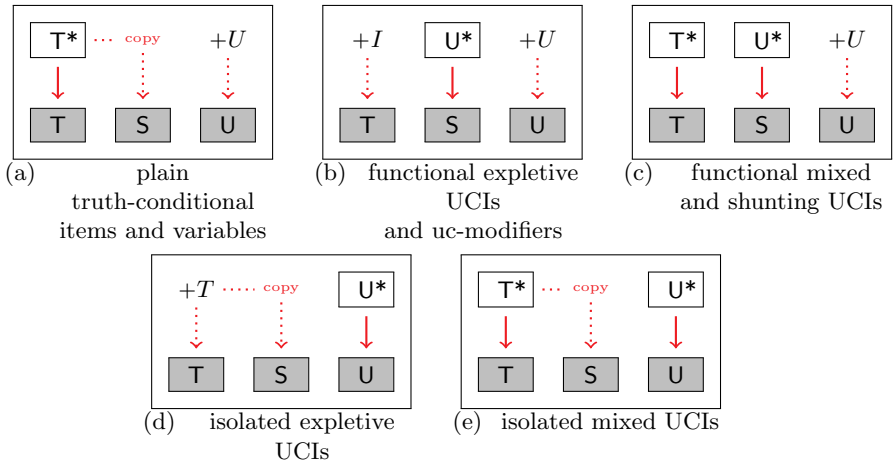
e. *expressive modification:*
$A : \langle \mathrm{T}, \mathrm{U} \rangle^n \Rightarrow I_{\mathrm{T}^n} \blacklozenge A : \langle \mathrm{T}, \mathrm{U} \rangle^n \bullet U$

f. *variables*:
$x_{\mathrm{T}} \Rightarrow x_{\mathrm{T}} \blacklozenge {}_{\mathrm{T}} \bullet U$

This set of LERs does us now allow to state the following hypothesis about the lexicon, which they link to the derivational semantics.

(23) *Hypothesis L^2*
The lexical entries are at most two-dimensional. They may encode up to one truth-conditional and up to one use-conditional dimension.

According to this hypothesis, lexical entries can either encode (i) just truth-conditional content, (ii) just use-conditional content, or (iii) both. Crucially, they cannot have two different use-conditional dimensions, in contrast to the 3-dimensional objects produced by the LERs, which distinguish between the s-dimension and the u-dimension. Intuitively, this makes sense, as the distinction between these two dimensions is rather a matter of the composition and not a question of different kinds of content. That the three dimensions do not line up perfectly with the conceptual different between truth- and use-conditional content is also shown by the fact that the s-dimension may also contain truth-conditional content. That is, the two dimensions that may be given by a lexical entry are not the same dimensions as the dimensions that we find in their 3-dimensional extensions. I call them therefore t*-dimension and u*-dimension respectively. The different ways in which these two *lexical dimensions* are distributed into the three *compositional dimensions* by the various LERs is illustrated in Fig. (23).



(a) plain truth-conditional items and variables (b) functional expletive UCIs and uc-modifiers (c) functional mixed and shunting UCIs (d) isolated expletive UCIs (e) isolated mixed UCIs

Fig. 6. Relation between lexical and compositional dimensions

7 Cross-dimensional quantification

With this system in place, we now only need an additional component to solve the problem of cross-dimensional quantification. What we need is a type shifter to transform an ordinary quantifier into a mixed quantifier that can apply to both dimensions.

(24) $$\frac{Q : \langle\langle e, t\rangle, t\rangle}{\star Q : \langle\langle e, u\rangle, u\rangle}$$ where $[\![\star Q]\!]$ = the function $f \in D_{\langle\langle e,u\rangle,u\rangle}$ such that for every $E \in D_{\langle e,u\rangle}$, $f(E) = \{c \in C : \{x \in D_e : E(x)(c) = 1\} \in [\![Q]\!]\}$.

With this type shifter in place, we can finally provide a derivation of the problematic case.

(25)

$$\dfrac{\dfrac{\dfrac{\dfrac{\textit{Everybody}}{\mathbf{evby} : \langle\langle e,t\rangle,t\rangle}}{\begin{array}{c}\blacklozenge\ \star\,\mathbf{evby} : \langle\langle e,u\rangle,u\rangle \bullet U \\ \mathbf{evby} : \langle\langle e,t\rangle,t\rangle\end{array}} \quad \dfrac{\dfrac{\textit{googles at}}{\begin{array}{c}\blacklozenge\ \lambda x\lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(x)(y)) : \langle e,\langle e,u\rangle\rangle \bullet U \\ \mathbf{look\text{-}at} : \langle e,\langle e,t\rangle\rangle\end{array}} \quad \dfrac{\dfrac{\dfrac{\dfrac{\textit{that damn}}{\mathbf{bad} : \langle e,u\rangle}}{\begin{array}{c}\bullet U \\ \blacklozenge\ \mathbf{bad} : \langle e,u\rangle \\ I_e\end{array}} \quad \dfrac{\dfrac{\textit{Klaus}}{\mathbf{k} : e}}{\begin{array}{c}\bullet U \\ \blacklozenge\ \mathbf{k} : e \\ \mathbf{k} :\end{array}}}{\mathbf{k} : e \blacklozenge \mathbf{bad}(\mathbf{k}) : u \bullet U}}{\mathbf{k} : e \blacklozenge \mathbf{k} : e \bullet \mathbf{bad}(\mathbf{k}) : u}}{\mathbf{look\text{-}at}(\mathbf{k}) : \langle e,t\rangle \blacklozenge \lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(\mathbf{k})(y))\,\langle e,u\rangle \bullet \mathbf{bad}(\mathbf{k}) : u}}{\mathbf{evby}(\mathbf{look\text{-}at}(\mathbf{k})) : t \blacklozenge \star\mathbf{evby}(\lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(\mathbf{k})(y)))u \bullet \mathbf{bad}(\mathbf{k}) : u}}{\mathbf{evby}(\mathbf{look\text{-}at}(\mathbf{k})) : t \blacklozenge \mathbf{evby}(\mathbf{look\text{-}at}(\mathbf{k})) : t \bullet \mathbf{bad}(\mathbf{k}) : u \odot \star\mathbf{evby}(\lambda y.\mathbf{bad}(\mathbf{look\text{-}at}(\mathbf{k})(y)))u}$$

8 Conclusion

We presented a reformulation of $\mathcal{L}^*_{\mathrm{CI}}$ that exhibits compositional multidimensionality, which enabled us to use just 2 composition rules and two kinds of types, while also solving the quantification problem. Of course, the complexity of the data must be accounted for somewhere and in the new system, this is done by lexical extension rules that serve as the bridge between a at most 2-dimensional lexicon and the 3-dimensional derivational semantic system.

References

[1] Potts, C.: The logic of conventional implicature. Oxford: Oxford University Press (2005)

[2] McCready, E.: Varieties of conventional implicature. Semantics & Pragmatics 3.8, 157 (2010)

[3] Gutzmann, D.: Expressive modifiers & mixed expressives. Empirical Issues in Syntax and Semantics 8, 123141 (2011)

[4] Potts, C.: The expressive dimension. Theoretical Linguistics 33.2, 165197 (2007)

[5] Gutzmann, D.: Use-conditional meaning. Studies in multidimensional semantics. PhD thesis. University of Frankfurt (2012)

[6] McCready, E., Takahashi, Y.: Good Reasons. In: Gutzmann, D., Grtner, H.-M.: Beyond expressives: explorations in use-conditional meaning. Brill: Leiden, 203-231 (2013)

[7] Frey, W.: A'-Movement and conventional implicatures: about the grammatical encoding of emphasis in German. Lingua 120, 14161435 (2010)

[8] Gutzmann, D.: Expressives and beyond. An introduction to varieties of use-conditional meaning. In: Gutzmann, D., Grtner, H.-M.: Beyond expressives: explorations in use-conditional meaning. Brill: Leiden, 1-58 (2013)

[9] Castroviejo Miro, E.: An expressive answer. Some considerations on the semantics and pragmatics of wh-exclamatives. Proceedings of CLS 44.2, 3-17 (2008)

[10] Barker, C., Bernardi, R., Shan, Ch.-ch.: Principles of interdimensional meaning interaction. Proceedings of SALT 20, 109121 (2010)

[11] Giorgolo, G., Asudeh, A.: $\langle M, \nu, \star \rangle$. Monads for conventional implicatures. Proceedings of Sinn und Bedeutung 16, 265-278 (2012)

[12] Heim, I., Kratzer, A.: Semantics in generative grammar. Oxford: Blackwell (1998)