1. Introduction

What are meaning shifts?

- We have a constituent \([\alpha, \beta]\) that, following syntax/semantic rules, would be interpreted as \(\alpha\beta\).
- This application fails for, type-theoretic reasons, or reasons related to world knowledge or expectation: \(*\alpha\beta\)
- The interpretation can be saved by first applying certain semantic shift operators \(Sh_1\) and/or \(Sh_2\): \(\text{Sh}(\alpha\beta)(\text{Sh}(\beta))\)

Examples: Metonymy


1. *The ham sandwich doesn’t like his ham sandwich.*
2. **Sh: \(\lambda y: \text{object} \ x \text{[person uniquely related to } y \text{ (in the given context)]}**

Notice:

- The function \(Sh\) is applicable in other cases as well, e.g.
  - a. The black BMW over there seems to be completely drunk.
  - b. The by-pass in room 202 needs immediate attention.
  - c. The bow-tie on table six refuses to pay.

- Use of a function \(Sh\) is pragmatically restricted, e.g. it helps the addressee identifying the object.

To be distinguished from lexical ambiguity:

3. Mary hit the bat hit the bat.

No available function that would map clubs to specimens of Chiroptera, or vice versa; we have to assume two distinct lexical entries.

Meaning shifts and idioms?

Such specialized shift operations have been assumed by Gazdar e.a. (1985) for transparent idioms (and hence are problematic, as shift operations):

4. \([\text{spill the beans}] = \text{Sh}(\text{spill})(\text{Sh}(\text{the beans})) = \text{reveal the secret}\)

Note that we would need a range of shift operators for *beans,* and *spill*:

5. a. She is full of beans. \(\text{Sh}(\text{beans})\) = spirited thoughts
   b. She doesn’t know beans about it. \(\text{Sh}(\text{bean})\) = minimal entity/amount
   c. blood was spilled

What does this mean for compositionality?

Just a mild extension (c):

6. The meaning of a complex expression \([\alpha, \beta]\) is a function of
   a. the meanings of its immediate syntactic parts \(\alpha, \beta\)
   b. the syntactic rule \(R\) of their combination,
   c. and shift operators \(Sh_1, Sh_2\) out of a finite (and small) set of operators that may have to be applied to the meanings of \(\alpha, \beta\).

Hence, meaning shifts need not be a “repair” mechanism after all.

The shift operators might be ranked according to complexity, and might be primed by context; the simplest “shift” operator being identity.

Overview of presentation:

- Type-theoretic meaning shifts (refereeing)
- Sortal meaning shifts (refereeing)
- Meaning shifts and complement variation (new)

I will concentrate here on more regular phenomena, and try to argue that seemingly irregular phenomena can be interpreted as following more general principles.

2. Type-theoretic meaning shifts

Shift operators are applied because \([\alpha, \beta]\) is not defined for type-theoretic reasons.

2.1 Generalized Boolean conjunction (and disjunction, negation)

Conjunction \(\land\) and disjunction \(\lor\) are Boolean operators, i.e. they are of type \(\langle\alpha, \beta, \beta\rangle\), mapping a pair of truth values to a truth value, but in natural language they can also be applied to expressions of other types.

7. a. John walked and Mary talked.
   b. John walked and talked.
   c. Every man and every woman walked.
   d. Mary was wearing a new and expensive dress.

Definition conjoinable type:

1. If \( x \) is a conjoinable type.
2. If \( \sigma \) is a type and \( \tau \) is a conjoinable type, then \( (\sigma)\tau \) is a conjoinable type.

Recursive definition of generalized conjunction:

1. If \( A \) Sh(\( A \)) B := A \land B if \( A, B \) of type \( t \).
2. If \( A \) Sh(\( A \)) B := \( \lambda x\{[A(X) \land B(X)] \} \) if \( A, B \) of type \( (\sigma)\tau \).

Example: Conjunction of predicates of type \( e \):

1. \( [\text{walk and talk}] = [\text{and}][[\text{walk}], [\text{talk}]] = * [\text{walk}] \land [\text{talk}] \)
2. After type shift:
   \( \text{Sh}[[\text{and}][[\text{walk}], [\text{talk}]]] = [\text{walk}][[\text{talk}]][x] \land [\text{talk}][x] \)
3. Application to subject:
   \( \lambda x[[\text{walk}][x] \land [\text{talk}][x]][\text{John}] = [[\text{walk}][\text{talk}][\text{John}]] \)

Complexity of this type shift in language acquisition: Bryant (2006)

(12) *Leo schenkte Hans ein Buch und Peter eine CD.*
   ‘Leo gave Hans a book and Peter a CD as a gift.’
   a. Adult preference: Leo gave Hans a book, and Leo gave Peter a CD: conjunction \( [\text{give Hans a book}] \) and \( [\text{give Peter a CD}] \)
   b. Child preference: Leo gave Hans a book and Peter gave Hans a CD: sentence conjunction: \( [\text{Leo gave Hans a book}] \) and \( [\text{Peter gave Hans a CD}] \)

Possible explanation:

- Adults: Preference for low attachment of […] Peter … eine CD]
- Children: Subject preference (here, for Peter)
- Children: Alternatively, preference for non-shifted conjunction, type \( t \).

Notice: We do not have to assume a shift operation at all; \( [\text{and}] = \text{Sh}(\text{A}) \).

2.2 Generalized sum formation

The basic case

English \( and \) has another reading, in addition to expressing Boolean \( \land \):

(13) \( \text{John and Mary kissed.} \neq \text{John kissed and Mary kissed.} \)

Proposal: Sum formation, \( \cup \), cf. e.g. Link (1983):

(14) \( \text{If} a, b \text{are individuals,} a \cup b \text{is an individual.} \)

Sum formation has certain mathematical properties and is related to a part relation \( \subseteq \) and overlap relation \( o \), see Champollion & Krifka (t.a.) for details.

(15) a. \( x \subseteq x \) Idempotency
    b. \( x \cup y = b \cup a \) Commutativity
    c. \( x \cup (y \cup z) = (x \cup y) \cup z \) Associativity

(16) a. \( x \subseteq y \iff x \cup y = y \) Part relation
    b. \( x \cup y = \exists z \{z \subseteq x \land z \subseteq y \} \) Overlap relation

Application to reciprocal predication:

(17) \( \text{[[[John and Mary’ kissed]]]} = \text{REC(kiss)(John} \cup \text{Mary)} \)

Meaning rule for REC operator, simplified (a) and generalized (b):

(18) a. \( \text{REC}(R)(x \cup y) :: R(x)(y) \land R(y)(x) \)
    b. \( \text{REC}(R)(x) :: R(x)(y) \land \forall y \{y \subseteq x \land \neg \forall z \{y \subseteq z \rightarrow R(y)(z)\} \}

Why do we use \( \land \) and \( \cup \)? Due to equivalence by distributive interpretation:

(19) \( \text{[[John and Mary walked]]} = \text{DIST}([[\text{walk}]])([[\text{John and Mary}]])
    = \lambda x\{x \subseteq x \land \exists y\{x \subseteq y \Land y \subseteq \text{walk}][\text{John}][\text{Mary}] \}
    = \{\text{walk}][\text{John}][\text{Mary}]

Sum formation generalized

Sum formation for predicates:

(20) a. \( \text{John and Mary [sang and danced]} \)
    b. \( \text{[boy and girl] who kissed each other} \)

Link (1983), extension of \( \cup \) from entities type \( e \) to predicates, type \( et \):

(21) \( \text{[P} \cup \text{P’]} = \lambda x\{x \subseteq x \land \psi (y) \Land P(y) \Land P’(z) \}

(22) \( \text{[The children [sang and danced]} = \{\text{sing and dance}][\text{the children]} \)
    = \lambda x\{x \subseteq x \Land \exists y\{y \subseteq y \Land \text{sing}(y) \Land \text{danc}(x)(c) \}
    = \lambda y\{y \subseteq y \Land \text{sing}(y) \Land \text{danc}(x)(c) \}

This is true if a part of the children sang and a part of the children danced, and the two parts make up all the children. This includes Boolean \( and \), as it might be that \( c=y \) and \( c=z \).

Sum formation applies also to other categories (Hoeksema 1988, Krifka 1990):

(23) a. \( \text{every boy and every girl kissed, type } (et) t \)
    b. \( \text{cheap and expensive dresses, type } (et) et \)
    c. \( \text{extremely and moderately expensive dresses, type } ((et)et)(et)et \)
    d. \( \text{the planes flew above the clouds and below the clouds, type } (et) et \)
    e. \( \text{the planes flew above and below the clouds, type } (et) et (e) et et \)

Generalized sum formation, Krifka 1990:

(24) Definition of \( e \)-conjoinable type:
   a. \( e \) is an \( e \)-conjoinable type;
   b. If \( \sigma \) is of an \( e \)-conjoinable type, then \( (\sigma) t \) is of an \( e \)-conjoinable type.
(25) Recursive definition of sum formation:
   a. \( \text{A Sh}(U) \) \( B = A \cup B \), if \( A, B \) are of type \( e \)
   b. \( \text{A Sh}(U) \) \( B = \lambda x_1 \exists y_1 \exists z_1 [X = Y \text{ Sh}(U) Z \land A(Y) \land B(Z)] \), if \( A, B \) of type \((e)t\)
   c. \( \text{A Sh}(U) \) \( B = \lambda y_{n_0} \ldots \lambda y_{n_m} [X_{n_0} = Y_{n_0} \ldots \lambda x_m = Y_m \text{ Sh}(U) Z_{n_0} \land \ldots \land \lambda x_m = Y_m \text{ Sh}(U) Z_{n_m} \land A(Y_{n_0} \ldots Y_{n_m}) \land B(Z_{n_0} \ldots B(Z_{n_m})) \)

(26) \[
\text{cheap} \text{ and expensive dresses} = \lambda \exists x \exists x' \exists x'' \left[ \text{cheap}(P)(x') \land \text{expensive}(P''(x'')) \right] \\
\text{cheap} \left[ \lambda \exists x \exists x' \exists x'' \left[ \text{cheap}(P)(x') \land \text{expensive}(P''(x'')) \right] \right] \\
\lambda \exists x \exists x' \exists x'' \left[ \text{cheap}(P)(x') \land \text{expensive}(P''(x'')) \right] \\
\lambda \exists x \exists x' \exists x'' \left[ \text{cheap}(P)(x') \land \text{expensive}(P''(x'')) \right] \\
\lambda \exists x \exists x' \exists x'' \left[ \text{cheap}(P)(x') \land \text{expensive}(P''(x'')) \right]
\]

2.3 NP/DP shifts: Partee 1987

Montague: NPs are all of quantifier type, \((e)t\) – generalizing to the most complex case.

(27) a. \( \forall x \text{man} = \lambda P \forall x [ \text{man}(x) \rightarrow P(x)] \)
   b. \( \exists x \text{man} = \lambda P \exists x [ \text{man}(x) \land P(x)] \)
   c. \( \text{John} = \lambda P (P(\text{John})) \)

Arguments against assuming a common type:

(28) a. John / the man / a man walked in. He looked tired.
   b. Every man / no man walked in. *He looked tired. (no coreference)

(29) Mary considers this an island / two islands / many islands / the prettiest island / the harbor / *every island / most islands / *Zanzibar.

Partee assumes that there shift operators that mediate between different types:

![Diagram of shift operators](image)

- Shift operators come in pairs
- Some of these shifts are restricted (partial functions): lower, iota, pred.

(30) Shifts in Chierchia (in extensional representations, for simplicity)
   a. \( \text{tiger} = \{x \mid x \text{ is a tiger}\} \)
   b. \( \text{Panthera_tigris} = \{x \mid x \text{ is a tiger}\} \)

(31) Predicate nouns:
   a. John is tall. \( \text{tall}(\text{John}) \)
   b. John is the mayor: \( \text{ident}([\text{the mayor}](\text{John})) \)

But shifts have to be restricted:

(32) a. Mary considers this an island.
   b. *Mary considers this Zanzibar.

Reason: Making shift explicit with Mary considers this to be Zanzibar?

2.4 DP shifts: Chierchia 1998, Krifka 2004, McNally & van Geenhoven

Chierchia (1998): Type shift analysis to explain versions of DPs in different languages.

(33) \[ \text{John and Mary walked} \]
   a. \( \text{DIST}([\text{walked}])([\text{John and Mary}]) \), cf. (19)
   b. \( \lambda P(\text{John}) \lambda P(\text{Mary}) \text{walked}([\text{walked}])([\text{walked}]) \)

If no agreement reflects non-atomicity, this suggests that (a) is preferred:

   b. Jeder Mann und jede Frau *gingen / geht nach Hause.

Notice that conjunction of type-distinct DPs may trigger shifts:

(35) \[ \text{John and every woman} \]

Agreement pattern in German reflects Boolean conjunction – judgements?

(36) \( \text{John und jede Frau} *\text{gingen / ging nach Hause} \)

2.4 DP shifts: Chierchia 1998, Krifka 2004, McNally & van Geenhoven

Chierchia (1998): Type shift analysis to explain versions of DPs in different languages.

(37) a. \( \text{NP}[+/- \text{arg}]: \) NPs can / cannot be arguments,
   b. \( \text{NP}[+/- \text{pred}]: \) NPs can/cannot be predicates
Examples of derivations, English:

(39) a. Gold is a metal. \[ \text{metal}(\text{AU}) \]; mass nouns denote kinds.

b. Dodos are extinct. \[ \text{extinct}(\text{dodos}) \]

\[ = \text{extinct}(\text{PL}([\text{dodo}]) \]

bare plurals are cumulative, can be mapped to kinds

c. *Dodo is extinct. \[ \text{extinct}(\text{dodo}) \],

ill-formed, as \( [\text{dodo}] \) is not cumulative; restricted to atoms.

Blocking type shifts: Indefinite article

(40) a. Dodos crowded. \[ \exists x [\text{dodos}(x) \land \text{P}(x)] \]

\[ = \lambda P \exists x ([\text{dodos}(x)] \land \text{P}(x)]\]

existential type shift

b. *Dodo crowded. \[ \exists x [\text{dodos}(x) \land \text{P}(x)] \]

as English has indefinite article for singular

c. A dodo crowded. \[ a [\text{dodos}(x)] \]

\[ = \lambda P:\text{atomic} \lambda P \exists x ([P(x) \land \text{P}(x)]([\text{dodos}]) \land \text{crowed}(x)) \]

\[ = \lambda P \exists x ([\text{dodos}(x)] \land \text{P}(x)] \land \text{crowed}(x)) \]

d. Water splashed. \[ \exists y [\text{H}_2\text{O}(x)] \]

\[ = \lambda P \exists x ([\text{H}_2\text{O}(x) \land \text{P}(x)]) \]

\[ = \exists x [\text{dodos}(x) \land \text{P}(x)] \]

existing type shift not blocked, as indefinite article cannot be applied.

Motivation of shifts

Chierchia assumes a more complex rule for (40)(a), Derived Kind Predication; this ensures that bare plurals have narrow scope w.r.t. other operators (cf. Carlson 1977), in contrast to e.g. some dogs.

(41) a. Dogs are everywhere.

\[ \forall y [\text{place}(y) \rightarrow \exists x [\text{dodos}(x) \land \text{be-at}(y)(x)] \]

b. Some dogs are everywhere.

\[ \text{in addition: } \exists x [\text{dodos}(x) \land \forall y [\text{place}(y) \rightarrow \text{be-at}(y)(x)]) \]

DKP: If \( P \) applies to objects and \( k \) is a kind, then \( P(k) = \exists x [k(x) \land \text{P}(x)] \)

Krifka (2004) decomposed this ad-hoc rule into a series of simpler shifts:

(43) \[ \exists [\text{PL}([\text{dog}])] \]

… but argued also that the shifts \( \cup \cap \) are not warranted, as \( \exists \) can apply without them; economy of shifts.

Narrow-scope phenomena

To capture narrow scope of bare nouns, van Geenhoven (1998) proposed a shift on predicates instead:

(44) If \( V \) is a verbal predicate, then \( \exists y = \lambda P \exists x [V(x) \land \text{P}(x)] \)

(45) \[ [\text{Dodos crowded}] = \exists [\text{crowed}([\text{dodos})]]. \]

\[ = \lambda P \exists x ([\text{crowed}(x) \land \text{P}(x)][\text{dodos}]) \]

\[ = \exists x [\text{crowed}(x) \land \text{dodos}(x)x] \]

3. Sortal Meaning Shifts

Shift operators are applied because \( [\alpha([\beta])] \) is not defined for sortal reasons. This includes metonymy, which will not be considered here.

3.1 Mass nouns and Count nouns: Grinder and Packager

The Universal Grinder

Common assumption: Objects have to be distinguished from stuff, as they may have different properties (cf. Link 1983)

(46) a. This ring is new, but the gold it is made of is old.

b. This ring was made in the Netherlands, but the gold it is made of comes from Peru.

Assume an operator function \( G: \lambda y: \text{object } \iota x [\text{the stuff of} y] \), the “universal grinder” (Pelletier 1975).

Count nouns refer to objects (to single objects in singular); mass nouns refer to stuff.

Predicate or a syntactic rule that sortally requires stuff is applied to an object may trigger G:

(47) There was sausage all over the floor:

a. Singular bare noun as DP requires that bare noun refers to stuff:

\[ \exists y [\text{DP} [\text{beer}]] \]

\[ = \lambda P \exists x [\text{beer}(x) \rightarrow \text{object}(x)] \]

b. *\[ \text{DP} [\text{sausage}]](x) \rightarrow \text{object}(x)] \]

c. Shift: \( \text{Sh}([\text{DP} [\text{sausage}]](x), \text{Sh}(P) = \lambda P \exists y [\text{P}(y) \land \text{G}(y) = x] \)

The universal packager

(48) He ordered a beer.

Assume an operator \( \text{Pck} \):

(49) \( \text{Pck} = \lambda y: \text{packaged } t x: \text{object}[x \text{is the object that consists in one package of} y] \)

(50) a. *\[ \text{DP} [\text{DP} [\text{beer}]] \]

b. Shift: \( \text{Sh}([\text{DP} [\text{beer}]](x), \text{Sh}(P) = \lambda P \exists y [\text{P}(y) \land x = \text{Pck}(y)] \)

3.2 Aspectual Shifts

Moens & Steedman (1988) discuss phenomena that force the temporal interpretation from one aspectual type to another.
Aspectual types (e.g., Vendler 1957):

(51) a. States: \( \text{John was sick. / John likes Mary.} \)
     Test: No progressive.

     b. Activities: \( \text{John walked. / Mary drank wine.} \)
     Test: for a while

     c. Accomplishments: \( \text{John recovered. / John walked a mile.} \)
     Test: in an hour

     d. Achievements: \( \text{John arrived. / John met Mary.} \)
     Test: state

     e. Semelfactives: \( \text{John hiccupped.} \)
     Test: event

But certain operators can trigger a variety of shifts ("aspectual coercion")

(52) a. \( \text{Sue played the sonata in twelve minutes.} \) — \( \text{play the sonata} \) is an accomplishment
     b. \( \text{Sue played the sonata for three minutes.} \) — here: activity, partial playing
     c. \( \text{Sue played the sonata for seven hours.} \) — here: activity, repeated playing
     d. \( \text{Sue played the sonata for years.} \) — here: state, habitual playing.

Shift operators applied to event predicate \( P = \lambda e \[ \text{play(e) } \land \text{AG(e)=sue } \land \text{PAT(e) = the son.} \] \), which is a quantized predicate, i.e. \( P(e) \land P(e') \land e \neq e' \rightarrow \neg P(e \sqcup e') \), cf. Krifka 1989.

Notice that these are not strictly sortal shifts; for sortal treatment cf. Bach 1991.

(53) \( \text{PART}(P) = \lambda e \exists e' [e \sqsubseteq e' \land P(e')] \);  
     \( \text{PART}(P) \) is a culative predicate, i.e. \( \text{PART}(P)(e) \land \text{PART}(P)(e') \rightarrow \text{PART}(P)(e \sqcup e') / \)

(54) \( \text{ITER}(P) = \lambda e \exists e' \[ \text{ITER}(P)(e) \land \text{ITER}(P)(e') \rightarrow \text{ITER}(P)(e \sqcup e') \] \);  
     \( \text{ITER}(P) \) is cumulative.

(55) \( \text{HAB}(P) = \text{GEN}(s \text{ is a suitable situation, } \exists e \in s \land P(e)) \),  
     this is a state: In suitable situations, \( P \) tends to happen.

Shift operations in Moens & Steeman’s paper:

Evidence for aspectual shifts:

cf. work summarized in Pylkkänen & McElree 2005:

Sentences like (a) rejected twice as often as (b), cf. Todorova e.a. 2000; interpreted as a garden path phenomenon that leads to a reanalysis.

(56) a. \( \text{Even though Howard sent a large check to his daughter for many years, she refused to accept his money.} \)
     b. \( \text{Even though Howard sent large checks to his daughter for many years, ...} \)

Further experimental results (probe recognition task) by Piñango e.a. 2006 show a delayed slowing reaction when aspectual coercion is required. Neurolinguistic studies (MEG) also show the effort of shift operations, cf. Brennan & Pylkkänen 2008.

3.3 Lexical Shifts: Complement coercion

Pustejovsky (1991, 1995) investigates phenomena like the following one:

(57) a. \( \text{Mary enjoyed the novel.} \) \( \text{(reading / writing the novel)} \)
     b. \( \text{Sue began the novel.} \) \( \text{(reading / writing the novel)} \)

Qualia structure:

     b. Formal role: that which distinguishes the object within a larger domain.
     c. Orientation, Magnitude, Shape, Dimensionality, Color, Position
     d. Telic Role: purpose and function of the object.
     e. Agentive role: Creator, Artifact, Natural Kind, Causal Chain

Examples

(59) novel: a. Constitutive role: Artefact(x)
     b. Form: book(x), disk(x), data in e-reader(x), ...
     c. Telos: read(T, y, x) (where T: telic event)
     d. Agentive: artifact(x), write(T, y, x)

(60) dictionary: a. Telos: reference(P, x)
     b. Agentive: artifact(x), compile(T, y, x)

(61) \( \text{Sue began the novel.} \) \[ \text{[begin]} \) requires a change;
     if applied to an entity of the sort of novels, two possible changes are provided:
     a. by Telos: reading;
     b. by Agentive: writing;
     this leads to the appropriate coercion of the argument \( \alpha[novel(x) ] \) to
     ‘reading the novel’ or ‘writing the novel’

(62) \( \text{Sue began the dictionary.} \)
     As there is no telic Telos role, we have just the agentive reinterpretation.

Evidence for meaning shift / complement coercion cf. e.g. work by Pylkkänen.
4. Meaning Shift and Complement Variation

4.1 to know that and to know wh

The verb know embeds that-clauses and embedded questions (cf. Karttunen 1977).

(a) Mary knows that John has won.  
(b) Mary knows who has won.

Karttunen assumes two distinct meanings of know related by a meaning postulate (fn. 11).

We assume that questions denote sets of propositions (Karttunen: true propositions):

\[ [\text{that John has won}] = \lambda i [\text{John has won in } i] \]

\[ [\text{who has won}] = \lambda p \exists x [\text{person}(x) \land p = \lambda i [x \text{ has won in } i]] \equiv \lambda i [x \text{ has won in } i] \land x \in \text{person} \]

Interpretation of to know that and to know wh:

\[ [\text{know}_0([i]) = \lambda x \forall i' \in K(i)(x) \forall x(p)(i')] \]

\[ 'x \text{ knows that } p \text{ at } i, \text{ provided that } p \text{ is true at } i, \text{ iff} \]

\[ \text{for all indices } i' \text{ that are epistemically accessible for } x \text{ at } i, \text{ } p \text{ is true at } i'. \]

\[ [\text{know}_0([i]) = \lambda Q \lambda x \forall p \in Q[p(i) \rightarrow [\text{know}([i])(p)(x)]] \]

\[ 'x \text{ knows } \forall Q \text{ at } i \text{ iff for all propositions } p \text{ in } Q, \]

\[ \text{if } p \text{ is true at } i, x \text{ knows that } p \text{ at } i'. \]

Stronger interpretation: \[ [\text{know}_0([i]) = \lambda Q \lambda x [\ldots \land [\neg p(i) \rightarrow [\text{know}([i])(\neg p)(x)]]] \]

Other options:

- Shifting of question meaning (set of propositions)
- to a proposition meaning (conjunction of all true propositions in the set).

4.2 Clause embedding / question embedding by meaning shift

Karttunen assumes a meaning postulate that relates the two interpretations of know. But this is a far more general phenomenon: There are many verbs that show the behavior of know (i.e. they embed that-clauses or questions).

Karttunen’s groups of question-embedding verbs:

\[ \text{a. Verbs of retaining knowledge: know, be aware, recall, remember, forget} \]

\[ \text{b. Verbs of acquiring knowledge: learn, notice, find out, discover} \]

\[ \text{c. Verbs of communication: tell, show, indicate, inform, disclose} \]

\[ \text{d. Decision verbs: decide, determine, specify, agree on, control} \]

\[ \text{e. Verbs of conjecture: guess, predict, bet on, estimate} \]

\[ \text{f. Opinion verbs: be certain about, have an idea about, be convinced about} \]

\[ \text{g. Inquisitive verbs: ask, wonder, investigate, be interested in} \]

\[ \text{h. Verbs of relevance: matter, be relevant, be important, care, be significant} \]

4.3 The case of believe

Why believe does not embed questions.

The verb believe, though similar to proposition-embedding know, does not embed questions:

\[ [\text{believe}(i) = \lambda p \lambda x \forall i' \in K(i)(x)[p(i')]] \]

\[ [\text{know}(i) = \lambda p \lambda x \forall i' \in K(i)(x)[p(i')]] \]

Factive presupposition is satisfied in question-embedding know:

\[ [\text{believe}(i) = \lambda Q \lambda x \forall p \in Q[p(i) \rightarrow [\text{believe}][i](p)(x)]] \]

Pragmatic rule “Maximize presupposition” forces the choice of know in case p is known to be true; cf. e.g. choice of definite article over indefinite article if coreference is intended (Heim 1987):

\[ [\text{know}(i) = \lambda Q \lambda x \forall i' \in K(i)(x)[p(i')]] \]

And why believe sometimes does embed them:

Possible objection: (73)(b) feels more like a syntactic violation. But this cannot be the case:
Mary could not believe who has won the race.

Interestingly, (79) entails that Mary knows who won, but is astonished about it. Possible solution: cannot believe as an idiom meaning "be astonished about", showing a different subcategorization behavior from believe.

But a compositional analysis is possible, as (79) differs from (80) beyond the presuppositional issue:

(80) Mary could not know who has won.

Meaning of (80) at i; the predicate know presupposes that its object is true in the context world of evaluation, not in the worlds that are made accessible by the modal operator could.

This refers to the believes of the speaker.

Verbs of decision express that a proposition becomes true at an index, i.e. it is not true yet at an index; meaning 'be astonished about', presupposes that its object is true in the context world of evaluation, not in the worlds that are made accessible by the modal operator could.

(81) Mary could not know who has won.

This refers to the believes of the speaker.

Verbs of decision express that a proposition becomes true at an index, i.e. it is not true yet at an index; meaning 'be astonished about', presupposes that its object is true in the context world of evaluation, not in the worlds that are made accessible by the modal operator could.

As before, believe is blocked because know expresses the presupposition p(i).

Meaning of (79) at i; the modal operator refers to the believes of Mary.

This states that there is no world i’ compatible with Mary’s knowledge at the actual world i such that for every proposition p in Q that is true at i’ (and hence, that Mary should consider true), Mary actually believes at i that p is true. How is this possible? If p is new information acquired by Mary at i’ (cf. common locution I see it but I don’t believe it.)

4.4 Other blocking restrictions

Decision verbs

(83) a. The jury decided that Mary won.

b. The jury decided who won.

Verbs of decision express that a proposition becomes true at an index, i.e. it is not true yet at an index. Hence the shift rule cannot apply in this way:

(84) ∀p∈Q[p(i) → decide(i)(p)(jury)],

where decide(i)(p)(x) → ¬p(i) ∧ ∀i’>i[p(i’)], >: temporal order of indices

Hence we have to consider the tenses:

(85) a. The jury is deciding who is winning.

b. The jury is deciding who will win.

c. The jury decided who won.

d. The jury decided who would win.

Now the shifting rule can be applied, e.g. for (c):

(86) a. Q = λp∃x[person(x) ∧ p = λi∃i’<i[win(i’)(x)]]

b. ∀p∈Q[p(i) → ∃i’<i[decide(i’)(p)(jury)]]

∀x∈person[∃i’<i[win(i’)(x)] → ∃i’<i[decide(i’)(p)(jury)]]

Verbs of conjecture

(87) a. Mary guessed which horse won.

b. Mary guessed which horse would win.

Guessing means expressing one’s expectation:

(88) Mary guessed that Lucky will win.

For (87) a Mary must have made a correct guess; this is compatible with the shift rule.

For (87) b would is a modal expressing Mary’s expectation; also compatible with shift rule.

Bibliography


Linguistics and Philosophy.


Linguistics and Philosophy.


Linguistics and Philosophy.


Linguistics and Philosophy.


Linguistics and Philosophy.

