SemDial 2011: Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue, pages 140–148.
Los Angeles, California, 21–23 September 2011.

The TTR perceptron: Dynamic perceptual meanings and semantic coordination

Staffan Larsson
University of Gothenburg
Sweden
sl@ling.gu.se

Abstract

In this paper, a dynamic semantic approach to subsymbolic perceptual aspects of meaning is presented. We show how a simple classifier of spatial information based on the Perceptron can be cast in TTR (Type Theory with Records). Furthermore, we show how subsymbolic aspects of meaning can be updated as a result of observing language use in interaction, thereby enabling fine-grained semantic plasticity and semantic coordination.

1 Introduction

In dynamic semantics, meanings are context-update functions which take an input context and return an updated (output) context. In this paper, a dynamic semantic approach to subsymbolic perceptual aspects of meaning is presented. We show how a simple classifier of spatial information based on the Perceptron can be cast in TTR (Type Theory with Records). A large variety of linguistic phenomena related to logical/symbolic meaning have already been addressed within this framework. Consequently, the TTR perceptron indicates that TTR may be a useful framework for integrating subsymbolic aspects of meaning in a way which allows us to keep around the accumulated insights from formal semantics.

Furthermore, we show how subsymbolic aspects of meaning can be updated as a result of observing language use in interaction, thereby enabling fine-grained semantic plasticity and semantic coordination. This is done by modeling a simple language game between agents with a shared focus of attention, similar to the "guessing game" of Steels and Belpaeme (2005).

The main contribution of this paper is thus to show how semantic coordination in dialogue concerning subsymbolic and perceptual aspects of meaning can be incorporated with traditional formal semantics.

We will first introduce the notion of semantic coordination. Then, we briefly introduce the TTR framework. In the following section, we show how perceptrons can be represented in TTR and how this can be used for incorporating subsymbolic semantics into a dynamic semantic / information state update framework.

2 Semantic coordination

Two agents do not need to share exactly the same linguistic resources (grammar, lexicon etc.) in order to be able to communicate, and an agent's linguistic resources can change during the course of a dialogue when she is confronted with a (for her) innovative use. For example, research on alignment shows that agents negotiate domain-specific microlanguages for the purposes of discussing the particular domain at hand (Clark and Wilkes-Gibbs, 1986; Garrod and Anderson, 1987; Pickering and Garrod, 2004; Brennan and Clark, 1996; Healey, 1997; Larsson, 2007). We use the term *semantic coordination* to refer to the process of interactively coordinating the meanings of linguistic expressions.

Several mechanisms are available for semantic coordination in dialogue. These include corrective feedback, where one DP implicitly corrects the way an expression is used by another DP (Father's first utterance in the dialogue below, taken from Clark (2007)), as well as explicit definitions of meanings (Father's second utterance).

"Gloves" example:

- Naomi: mittens
- Father: **gloves**.
- Naomi: gloves.
- Father: when they have fingers in them they are called gloves and when the fingers are all put together they are called mittens.

It also possible to coordinate silently, by DPs observing the language use of others and adapting to it. Here's a modified version of the "gloves" example which we will use to illustrate this:

Modified "Gloves" example:

- (Naomi is putting on her new gloves)
- Father: Those are nice gloves!

In Larsson (2010) we sketch a formal account of learning meanings from observation and accommodation in dialogue. The examples we present are from first language acquisition, where the child detects innovative (for her) uses and adapts her take on the meaning accordingly. We regard semantic coordination in first language acquisition as a special case of semantic coordination in general, where there is a clear asymmetry between the agents involved with respect to expertise in the language being acquired when a child and an adult interact. However, we believe that the mechanisms for semantic coordination used in these situations are similar to those which are used when competent adult language users coordinate their language.

3 The left-or-right game

As an illustration, we will be using a simple language game whose objective is to negotiate the meanings of the words "left" and "right". A and B are facing a framed surface on a wall, and A has a bag of objects which can be attached to the framed surface. The following procedure is repeated:

1. A places an object in the frame
2. B orients to the new object, assigns it a unique individual marker and orients to it as the current object in shared focus of attention
3. A says either ”left” or ”right”
4. B interprets A's utterance based on B's take on the situation. Interpretation involves determining whether B's understanding of A's utterance is consistent with B's take on the situation.
5. If an inconsistency results from interpretation, B assumes A is right, says "aha", and learns from this exchange; otherwise, B says "okay"

Note that the resulting meanings of "left" and "right" will depend on how A places the objects in the frame and what A says when doing so; this may or may not correspond to the standard everyday meanings of "left" and "right". However, to keep things intuituve we will assume that A's takes on the meanings of these words can be paraphrased as "to the left of the center of the frame" and "to the right of the center of the frame", respectively.

The left-or-right game can be regarded as a considerably pared-down version of the "guessing game" in Steels and Belpaeme (2005), where perceptually grounded colour terms are learnt from interaction.

The kinds of meanings learnt in the left-or-right game may be considered trivial. However, at the moment we are mainly interested in the basic principles of combining formal dynamic semantics with learning of perceptual meaning from dialogue, and the hope is that these can be formulated in a general way which can later be used in more interesting settings.

The remainder of this paper will be spent formulating this simple game in TTR. To this end, we give a brief introduction to this framework.

4 TTR

We will take an information state update approach to utterance interpretation, using Type Theory with Records (TTR) to model contexts and meaning functions.

We can here only give a brief introduction to TTR; see also Cooper (2005) and Cooper (fthc). The advantage of TTR is that it integrates logical techniques such as binding and the lambda-calculus into feature-structure like objects called record types. Thus we get more structure than in a traditional formal semantics and more logic than is available in traditional unification-based systems. The feature structure like properties are important for developing similarity metrics on meanings and for the straightforward definition of meaning modifications involving refinement and generalization. The logical aspects are important for relating our semantics to the model and proof theoretic tradition associated with compositional semantics.

We will now briefly introduce the TTR formalism. If $a_1 : T_1, a_2 : T_2(a_1), \ldots, a_n : T_n(a_1, a_2, \ldots, a_{n-1})$, the record to the left is of the record type to the right:

$$\begin{bmatrix} l_1 & = & a_1 \\ l_2 & = & a_2 \\ \ldots & & \\ l_n & = & a_n \\ \ldots & & \end{bmatrix} : \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2(l_1) \\ \ldots & & \\ l_n & : & T_n(l_1, l_2, \ldots, l_{n-1}) \end{bmatrix}$$

Types constructed with predicates may also be *dependent*. This is represented by the fact that arguments to the predicate may be represented by labels used on the left of the ':' elsewhere in the record type.

Some of our types will contain *manifest fields* (Coquand et al., 2004) like the a-field in the following type:

$$\begin{bmatrix} \text{a=obj123} & : & \text{Ind} \\ \text{b} & : & \text{Ind} \end{bmatrix}$$

$\begin{bmatrix}\text{ref=obj123:Ind}\end{bmatrix}$ is a convenient notation for $\begin{bmatrix}\text{ref} : \text{Ind}_{\text{obj123}}\end{bmatrix}$ where $\text{Ind}_{\text{obj123}}$ is a *singleton type*. If $a : T$, then T_a is a singleton type and $b : T_a$ (i.e. b is of type T_a) iff $b = a$. Manifest fields allow us to progressively specify what values are required for the fields in a type.

An important notion in this kind of type theory is that of *subtype*. Formally, $T_1 \sqsubseteq T_2$ means that T_1 is a subtype of T_2. Two examples will suffice as explanation of this notion:

$$\begin{bmatrix} \text{ref} & : & \text{Ind} \\ \text{c} & : & \text{glove(ref)} \end{bmatrix} \sqsubseteq \begin{bmatrix} \text{ref} & : & \text{Ind} \end{bmatrix}$$

$$\begin{bmatrix} \text{ref=obj123} & : & \text{Ind} \end{bmatrix} \sqsubseteq \begin{bmatrix} \text{ref} & : & \text{Ind} \end{bmatrix}$$

Below, we will also have use for an operator for combining record types. The $\underset{.}{\wedge}$ operator works as follows. Suppose that we have two record types C_1 and C_2:

$$C_1 = \begin{bmatrix} \text{x} : \textit{Ind} \\ \text{c}_{\text{clothing}} : \text{clothing(x)} \end{bmatrix}$$

$$C_2 = \begin{bmatrix} \text{x} : \textit{Ind} \\ \text{c}_{\text{physobj}} : \text{physobj(x)} \end{bmatrix}$$

In this case, $C_1 \wedge C_2$ is a type. In general if T_1 and T_2 are types then $T_1 \wedge T_2$ is a type and $a : T_1 \wedge T_2$ iff $a : T_1$ and $a : T_2$. A meet type $T_1 \wedge T_2$ of two record types can be simplified to a new record type by a process similar to unification in feature-based systems. We will represent the simplified type by putting a dot under the symbol $\wedge$. Thus if T_1 and T_2 are record types then there will be a type $T_1 \underset{.}{\wedge} T_2$ equivalent to $T_1 \wedge T_2$ (in the sense that a will be of the first type if and only if it is of the second type).

$$C_1 \underset{.}{\wedge} C_2 = \begin{bmatrix} \text{x} : \textit{Ind} \\ \text{c}_{\text{physobj}} : \text{physobj(x)} \\ \text{c}_{\text{clothing}} : \text{clothing(x)} \end{bmatrix}$$

The operation $\underset{.}{\wedge}$, referred to as *merge* below, corresponds to unification in feature-based systems and its definition (which we omit here) is similar to the graph unification algorithm.

5 Dynamic subsymbolic semantics

In this section, we will show how a TTR-based dynamic semantic account of meaning can be extended to incorporate subsymbolic aspects of meaning. Examples will be based on the left-or-right game introduced above.

5.1 Perceptual meanings as classifiers

Since all aspects of meaning can be modified as a result of language use in dialogue, we want our account of semantic coordination and semantic plasticity to include several aspects of lexical meaning. We take the lexical meaning $[e]$ of an expression e to contain not only compositional semantics but also perceptual meaning. By this we mean that aspect of the meaning of an expression which allows an agent to detect objects or situations referred to by the expression e. For example, knowing the perceptual meaning of "panda" allows an agent to correctly classify pandas in her environment as pandas.

Likewise, an agent which is able to compute the perceptual meaning of "a boy hugs a dog" will be able to correctly classify situations where a boy hugs a dog. We can therefore think of perceptual meanings as classifiers of sensory input.

5.2 The TTR perceptron

Classification of perceptual input can be regarded as a mapping of sensor readings to types To represent perceptual classifiers, we will be using a simple perceptron. A perceptron is a very simple neuron-like object with several inputs and one output. Each input is multiplied by a weight and if the summed inputs exceed a threshold, the perceptron yields as output, otherwise 0 (or in some versions -1).

$$o(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} > \mathrm{t} \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^{n} w_i x_i = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$

Perceptrons are limited to learning problems which are linearly separable; the distinction between left and right is one such problem. Perceptrons can be interconnected by connecting the output of one or several perceptrons to the inputs of a different perceptron. Also, perceptrons can also be used to model reasoning. Here, we want to use a single perceptron to model perception.

In TTR, an n-dimensional real-valued vector will be represented as a record with labels $1, \ldots, n$ where the value of each label will be a real number. Such a records will be of the type $\mathrm{RealVector}_n$.

$$\mathrm{RealVector}_n = \begin{bmatrix} 1 & : & \mathrm{Real} \\ 2 & : & \mathrm{Real} \\ \ldots & & \\ \mathrm{n} & : & \mathrm{Real} \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} 1 & = & 0.23 \\ 2 & = & 0.34 \\ 3 & = & 0.45 \end{bmatrix} : \mathrm{RealVector}_3$$

For convenience, we will abbreviate this as in this example:

$$\mathbf{x} = \begin{bmatrix} 0.23 & 0.34 & 0.45 \end{bmatrix} : \mathrm{RealVector}_3$$

$\mathbf{x}_n = \mathbf{x}.\mathrm{n}$, so $\mathbf{x}_2$=0.34

5.3 The TTR perceptron as a classifier

The basic perceptron returns a real-valued number (1.0 or 0.0) but when we use a perceptron as a classifier we want it to instead return a type. Typically, such types will be built from a predicate and some number of arguments; for the moment we can think of this type as a "proposition".

A TTR classifier perceptron for a type P can be represented as a record:

$$\begin{bmatrix} \mathrm{w} & = & \begin{bmatrix} 0.800 & 0.010 \end{bmatrix} \\ \mathrm{t} & = & 0.090 \\ \mathrm{fun} & = & \lambda v : \mathrm{RealVector} \\ & & (\begin{cases} P & \text{if } v \cdot \mathrm{w} > \mathrm{t} \\ \neg P & \text{otherwise} \end{cases}) \end{bmatrix}$$

Where p.fun will evaluate to

$$\lambda v : \mathrm{RealVector} \\ (\begin{cases} \mathrm{P} & \text{if } v \cdot \begin{bmatrix} 0.100 & 0.200 \end{bmatrix} > 0.090 \\ \neg\, \mathrm{P} & \text{otherwise} \end{cases})$$

5.4 Situations and sensors

In first language acquisition, training of perceptual classifiers typically takes place in situations where the referent is in the shared focus of attention and thus perceivable to the dialogue participants, and for the time being we limit our analysis to such cases. For our current purposes, we assume that our DPs are able to establish a shared focus of attention, and we will designate the label foc-obj for the object or objects taken by a DP to be in shared focus.

A (simple) sensor collects some information (sensor input) from the environment and emits a real-valued vector. The sensor is assumed to be oriented towards the object in shared focus.

An agent's (possibly underspecified) take on a situation is a part of the agent's information state. It is represented as a record type, possibly containing manifest fields.

Furthermore, we will assume that sensors are directed towards the focused object.

In the left-or-right game, we will assume that B's take on the situation includes readings from a position sensor (denoted "sr_{pos}") and a field foc-obj for an object in shared focus of attention. The position sensor returns a two-dimensional real-valued vector representing the horizontal vertical coordinates of the focused object: $\begin{bmatrix}x & y\end{bmatrix}$ where $-1.0 \leq x, y \leq 1.0$ and $\begin{bmatrix}0.0 & 0.0\end{bmatrix}$ represents the center of the frame.

Here is an example of B's take on the situation prior to playing a round of the left-or-right game:

$$s_1^B = \begin{bmatrix} \text{sr}_{pos}{=}\begin{bmatrix}0.900 & 0.100\end{bmatrix} & : & \text{RealVector} \\ \text{foc-obj=obj}_{45} & : & \text{Ind} \\ \text{spkr=A} & : & \text{Ind} \end{bmatrix}$$

In s_1^b, B's sensor is oriented towards obj_{45} and sr_{pos} returns a vector corresponding to the position of obj_{45}.

5.5 Sensors readings as proofs

A fundamental type-theoretical intuition is that something of type $P(a)$ is whatever it is that counts as a proof that P holds of a. One way of putting this is that "propositions are types of proofs". One can have different ideas of what kind of objects count as proofs. Here we will be assuming that proof-objects can be *takes on situations* involving *readings from sensors*; we can call such a proof a *verification*.

5.6 Static and dynamic semantics

We will take parts of the meaning of an uttered expression to be *foregrounded*, and other parts to be *backgrounded*. Background meaning (bg) represents constraints on the context, whereas foreground material (fg) is the information to be added to the context by the utterance in question. Both background and foreground meaning components are represented in TTR as types:

bg = T_1
fg = T_2

Static meanings (Kaplans "character") are functions from records (representing situations) to record types (representing Kaplan's "content").

$\lambda x : T_1(T_2)$

In TTR, contexts are represented as records, whereas an agent's *takes* on a context is represented as a record type (typically involving manifest fields). This allows takes on contexts to be underspecified, which is useful in modeling agents with incomplete knowledge. In TTR dynamic semantics, we therefore need a different kind of function, namely one which takes an agents take on the context as input and returns an updated take on the context. In TTR terms, this means we need a function from record types to record types.

$\lambda x \sqsubseteq T_1(x \dot{\wedge} T_2)$

When representing the meaning of an expression e in lexicon, we can use a record collecting the various aspects of $[e]$, the meaning of e:

$$[\text{e}] = \begin{bmatrix} \text{bg} & = & T_1 \\ \text{fg} & = & T_2 \\ \text{sfun} & = & \lambda x : \text{bg}(\text{fg}[\text{bg}/x]) \\ \text{dfun} & = & \lambda x \sqsubseteq \text{bg}(x \dot{\wedge} \text{fg}[\text{bg}/x]) \end{bmatrix}$$

where $e_1[e_2/e_3]$ is e_1 with any occurrences of e_2 replaced by e_3.

The context update function (dfun, where d is for "dynamic" as in "dynamic semantics") takes as argument a record type x (representing the agent's take on a situation) which is a subtype of the background meaning of the uttered expression, and returns a record type corresponding to the merge of x and the foreground meaning. Dynamic contextual interpretation amounts to applying this function to the input context, and the result of function application is the output context[1].

As it happens, there are several things which may go wrong in interpreting an utterance. Given our formulation of the context update function corresponding to meanings, we can describe three cases of mismatch between context c and the meaning of an expression e:

1. **Type mismatch**: The input context is not a subtype of the background meaning: $c \not\sqsubseteq [e].\text{bg}$
2. **Background inconsistency**: The input context is inconsistent with background meaning: $[e].\text{bg} \dot{\wedge} c \approx \bot$
3. **Foreground inconsistency**: The output context is inconsistent[2]: $[e]@(c) \approx \bot$

In the following, we will focus on foreground inconsistency, leaving the other cases for future work.

5.7 The meaning of "right"

We can now say what a meaning in B's lexicon might look like before a round of the left-or-right game. We assume that B has meanings only for "left" and

[1]Note that we will be using "context" and "situation" interchangeably.

[2]We use @ to denote function application.

$$[\text{right}]^B = \begin{bmatrix} \text{w} = \begin{bmatrix} 0.800 & 0.010 \end{bmatrix} \\ \text{t} = 0.090 \\ \text{bg} = \begin{bmatrix} \text{sr}_{pos} & : & \text{RealVector} \\ \text{foc-obj} & : & \text{Ind} \\ \text{spkr} & : & \text{Ind} \end{bmatrix} \\ \text{fg} = \begin{bmatrix} \text{c}^{perc}_{right} = \begin{bmatrix} \text{sr}_{pos} = \text{bg.sr}_{pos} \\ \text{foc-obj} = \text{bg.foc-obj} \end{bmatrix} : \begin{cases} \text{right(bg.foc-obj)} & \text{if bg.sr}_{pos}\cdot\text{w} > \text{bg.sr}_{pos}\cdot\text{t} \\ \neg\text{right(bg.foc-obj)} & \text{otherwise} \end{cases} \\ \text{c}^{tell}_{right} = \begin{bmatrix} \text{str} = \text{“right”} \\ \text{spkr} = \text{bg.spkr} \\ \text{foc-obj} = \text{bg.foc-obj} \end{bmatrix} : \text{right(bg.foc-obj)} \end{bmatrix} \\ \text{sfun} = \lambda x : \text{bg(fg[bg}/x\text{])} \\ \text{dfun} = \lambda x \sqsubseteq \text{bg(}x \dot{\wedge} \text{fg[bg}/x\text{])} \end{bmatrix}$$

Figure 1: B’s initial lexical entry for ”right”

“right”.In our representations of meanings, we will combine the TTR representations of meanings with the TTR representation of classifier perceptrons.

Agent B’s initial take on the meaning of “right” is represented by the record in Figure 1. The fields w and t specify weights and a threshold for a classifier perceptron which is used to classify sensor readings.

The bg field represents constraints on the input context, which requires that there is a colour sensor reading and a focused object foc-obj. The fg field specifies two fields.

The value of c^{perc}_{right} is a proof of either or right(foc-obj) or ¬right(foc-obj), depending on the output of the classifier perceptron which makes use of w and t. Here, right(y) is a perceptual “proposition” (a type constructed from a predicate), and objects of this type are proofs that y is (to the) right. As a proof of right(foc-obj) we count a “snapshot” of relevant parts of the situation, consisting of the current sensor reading and a specification of the currently focused object.

The value of c^{tell}_{right} is a record containing information about an utterance, namely that a speaker just uttered the word “right”. We assume that this counts as a proof that foo is (to the) right. This implements an assumption that A is always right, an assumption that one could choose to remove in a more complicated version of the left-or-right game.

6 Contextual interpretation

We will first show a case where interpretation runs smoothly. Player A picks up an object and places it in the frame, and B finds the object and assigns it the individual marker obj$_{45}$, directs the position sensor to it and gets a reading. Player A now says “right”, after which B’s take on the situation is s^B_1, repeated here for convenience:

$$\text{s}^B_1 = \begin{bmatrix} \text{sr}_{pos}=\begin{bmatrix} 0.900 & 0.100 \end{bmatrix} & : & \text{RealVector} \\ \text{foc-obj=obj}_{45} & : & \text{Ind} \\ \text{spkr=A} & : & \text{Ind} \end{bmatrix}$$

To interpret A’s utterance, B applies $[\text{right}]^B$.dfun to s^B_1 to yield a new take on the situation s^B_2:

$\text{s}^B_2=[\text{right}]^B\text{.dfun@s}^B_1$ =

$$\begin{bmatrix} \text{sr}_{pos}=\begin{bmatrix} 0.900 & 0.100 \end{bmatrix}\text{:RealVector} \\ \text{foc-obj=obj}_{45}\text{:Ind} \\ \text{spkr=A:Ind} \\ \text{c}^{perc}_{right} = \begin{bmatrix} \text{sensor}_{col} = \begin{bmatrix} 0.900 & 0.100 \end{bmatrix} \\ \text{foc-obj} = \text{obj}_{45} \end{bmatrix} \text{:right(obj}_{45}\text{)} \\ \text{c}^{tell}_{right} = \begin{bmatrix} \text{str} = \text{“right”} \\ \text{spkr} = \text{A} \\ \text{foc-obj} = \text{obj}_{45} \end{bmatrix} \text{:right(obj}_{45}\text{)} \end{bmatrix}$$

Here, the classifier takes s^B_1 to contain a proof of right(obj$_{45}$).

7 Learning perceptual meaning from interaction

7.1 Detecting foreground inconsistency

We now assume that in the next round, A places another object in a different position in the frame and again says “right”. Now, B’s take on the situation is as follows:

$$
\mathrm{s}_3^B = \left[\begin{array}{l}
\mathrm{sr}_{pos}=\begin{bmatrix}0.100 & 0.200\end{bmatrix}:\mathrm{RealVector} \\
\text{foc-obj}=\mathrm{obj}_{45}:\mathrm{Ind} \\
\mathrm{spkr}=\mathrm{A}:\mathrm{Ind} \\
\mathrm{c}_{right}^{perc} = \left[\begin{array}{l}\mathrm{sensor}_{col} = \begin{bmatrix}0.900 & 0.100\end{bmatrix} \\ \text{foc-obj} = \mathrm{obj}_{46}\end{array}\right] : \mathrm{right}(\mathrm{obj}_{45}) \\
\mathrm{c}_{right}^{tell} = \left[\begin{array}{l}\mathrm{str} = \text{“right”} \\ \mathrm{spkr} = \mathrm{A} \\ \text{foc-obj} = \mathrm{obj}_{45}\end{array}\right] : \mathrm{right}(\mathrm{obj}_{45})
\end{array}\right]
$$

Note that foc-obj has been updated and that there is a new sensor reading[3]. As before, B interprets A’s utterance to yield a new take on the situation[4]:

$$
\mathrm{s}_4^B = [\mathrm{right}]^B.\mathrm{dyn}@\mathrm{s}_3^B = \left[\begin{array}{l}
\mathrm{sr}_{pos}=\begin{bmatrix}0.100 & 0.200\end{bmatrix}:\mathrm{RealVector} \\
\text{foc-obj}=\mathrm{obj}_{45}:\mathrm{Ind} \\
\mathrm{spkr}=\mathrm{A}:\mathrm{Ind} \\
\mathrm{c}_{right}^{perc} = \left[\begin{array}{l}\mathrm{sensor}_{col} = \begin{bmatrix}0.900 & 0.100\end{bmatrix} \\ \text{foc-obj} = \mathrm{obj}_{45}\end{array}\right] : \mathrm{right}(\mathrm{obj}_{45}) \\
\mathrm{c}_{right}^{tell} = \left[\begin{array}{l}\mathrm{str} = \text{“right”} \\ \mathrm{spkr} = \mathrm{A} \\ \text{foc-obj} = \mathrm{obj}_{45}\end{array}\right] : \mathrm{right}(\mathrm{obj}_{45}) \\
\mathrm{c1}_{right}^{perc} = \left[\begin{array}{l}\mathrm{sensor}_{col} = \begin{bmatrix}0.100 & 0.200\end{bmatrix} \\ \text{foc-obj} = \mathrm{obj}_{46}\end{array}\right] : \neg\,\mathrm{right}(\mathrm{obj}_{46}) \\
\mathrm{c1}_{right}^{tell} = \left[\begin{array}{l}\mathrm{str} = \text{“right”} \\ \mathrm{spkr} = \mathrm{A} \\ \text{foc-obj} = \mathrm{obj}_{46}\end{array}\right] : \mathrm{right}(\mathrm{obj}_{46})
\end{array}\right]
$$

This time, however, applying the classifier perceptron to the sensor input yields $\neg$right(obj$_{46}$) and hence the classifier takes s_3^B to contain a proof both of right(obj$_{46}$) (labelled $\mathrm{c1}_{right}^{perc}$) and of $\neg$right(obj$_{46}$) (labelled $\mathrm{c1}_{right}^{tell}$). This is a case of foreground inconsistency – the record type s_4^B is inconsistent ($\mathrm{s}_4^B \approx \perp$). That is, there can be no situation (record) of this type.

According to the rules of the game, B resolves this conflict by trusting A’s judgement over B’s own classification. Hence, B must remove $\mathrm{c1}_{right}^{perc}$. Furthermore, B can learn from this exchange by updating the weights used by the classifier perceptron associated with [right].

[3]We are assuming that takes on situations can be updated not only by applying dynamic meanings to them, but also by applying non-monotonic updates, as in the Information State Update approach to dialogue management (Traum and Larsson, 2003). Specifically, we assume the values of sr$_{pos}$, foc-obj and spkr have been updated in this way.

[4]We are assuming a mechanism for relabeling fields if labels conflict, by attaching an integer to the label, starting with 1.

7.2 Updating perceptual meaning

Perceptrons are updated using the *perceptron training rule*:

$$w_i \leftarrow w_i + \Delta w_i$$

where

$$\Delta w_i = \eta(o_t - \mathrm{o})x_i$$

where o_t is the target output, o is the actual output, and w_i is associated with input x_i. Note that if $o_t = o$, there is no learning. However, since B only tries to learn from mistakes (and not from successes), we have already established that $o_t - o$ is 1.0 for a perceptron outputting real numbers. (In any case, our classifier perceptron instead outputs types, so it is not very useful for computing this difference.)

We can now formulate the perceptron training rule as updating a TTR record[5] :

ptrain(m, s, C) = m but with
  m.w$\leftarrow$ m.w+$\eta^n \cdot s$.sr$_C$
  m.t$\leftarrow$ m.t$-\eta$

where

- m is a meaning (e.g. [right])
- s is a record type representing a take on a situation
- C is a sensor name, e.g. *pos*, corresponding to a perceptual category (e.g. position)
- m.w:RealVector$_n$, m.t:Real
- η^n is an n-dimensional real-valued vector where $\eta^n_m = \eta$ for all m, $1 \leq m \leq n$, e.g. $\eta^2 = \begin{bmatrix}\eta & \eta\end{bmatrix}$
- s.sr$_C$ is a sensor reading in s

[5]We here train the threshold t separately; an alternative is to include it as w$_0$ and assume a dummy input x$_0$ = 1. In the latter case, t is updated as a part of updating w.

In the example above, for $\eta = 0.1$ we get

[right]^{B}' = ptrain([right]^{B}, s_4^B, *pos*) = [right]^{B} but with [right]^{B}.w←

$$\begin{bmatrix}0.800 & 0.010\end{bmatrix} + \begin{bmatrix}0.1 & 0.1\end{bmatrix} \cdot \begin{bmatrix}0.100 & 0.200\end{bmatrix} \text{ and}$$

$$m.\text{t} \leftarrow 0.090 - 0,1$$

which yields

$$[\text{right}]^{B}\text{'.w} = \begin{bmatrix}0.808 & 0.2002\end{bmatrix}.$$
$$[\text{right}]^{B}\text{'.t} = -0.010.$$

B has thus updated the meaning of "right" by modifying the weight vector used by a classifier perceptron, based on the output of applying the dynamic semantics of "right" to B's take on the situation.

8 Conclusion and future work

The work presented here is part of a research agenda aiming towards a formal account of semantic coordination in dialogue. In this paper, we have presented a dynamic semantic approach to subsymbolic perceptual aspects of meaning. We have shown how a simple classifier of spatial information based on the Perceptron can be cast in TTR (Type Theory with Records). Furthermore, we have shown how subsymbolic aspects of meaning can be updated as a result of observing language use in interaction, thereby enabling fine-grained semantic plasticity and semantic coordination.

There are many possible variants of the left-or-right game, which will be explored in future research. An obvious extension is to add more words (e.g. "upper" and "lower") and some simple grammar ("upper left", "lower right" etc) to explore compositionality of perceptual meanings. The left-or-right game can be extended by adding more interesting interaction patterns, including corrective feedback and explicit definitions. The capabilities of the agents could be extended by e.g. pointing. Additional sensors and classifiers, e.g. for colour, shape and relative position, can be added. The fact that situations are stored as proofs can be useful in interactions where agent B rejects an utterance of by A and cites a previous situation when arguing for this rejection ('If this one here [pointing at object] was on the right, how can this one [pointing at other object] be on the left?'). We also want to explore how cases of type mismatch and background inconsistency can play out in (some more sophisticated version of) the left-or-right game.

Acknowledgments

This research was supported by The Swedish Bank Tercentenary Foundation Project P2007/0717, Semantic Coordination in Dialogue. Thanks to Robin Cooper and Simon Dobnik for extremely useful comments on an early draft.

References

Brennan, S. E. and H. H. Clark (1996). Conceptual pacts and lexical choice in conversation. *Journal of Experimental Psychology: Learning, Memory and Cognition 22*, 482–493.

Clark, E. V. (2007). Young children's uptake of new words in conversation. *Language in Society 36*, 157–82.

Clark, H. H. and D. Wilkes-Gibbs (1986). Refering as a collaborative process. *Cognition 22*, 1–39.

Cooper, R. (2005). Austinian truth, attitudes and type theory. *Research on Language and Computation 3*, 333–362.

Cooper, R. (fthc). Type theory and semantics in flux.

Coquand, T., R. Pollack, and M. Takeyama (2004). A logical framework with dependently typed records. *Fundamenta Informaticae XX*, 1–22.

Garrod, S. C. and A. Anderson (1987). Saying what you mean in dialogue: a study in conceptual and semantic co-ordination. *Cognition 27*, 181–218.

Healey, P. (1997). Expertise or expertese?: The emergence of task-oriented sub-languages. In M. Shafto and P. Langley (Eds.), *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, pp. 301–306.

Larsson, S. (2007). Coordinating on ad-hoc semantic systems in dialogue. In *Proceedings of the 10th workshop on the semantics and pragmatics of dialogue*.

Larsson, S. (2010). Accommodating innovative meaning in dialogue. In P. Łupkowski and M. Purver (Eds.), *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dia-*

logue, pp. 83–90. Poznań: Polish Society for Cognitive Science.

Pickering, M. J. and S. Garrod (2004, April). Toward a mechanistic psychology of dialogue. *Behavioral and Brain Sciences 27*(02), 169–226.

Steels, L. and T. Belpaeme (2005, August). Coordinating perceptually grounded categories through language: A case study for colour. *Behavioral and Brain Sciences 28*(4), 469–89. Target Paper, discussion 489-529.

Traum, D. and S. Larsson (2003). The information state approach to dialogue management. In R. Smith and J. Kuppevelt (Eds.), *Current and New Directions in Discourse & Dialogue*. Kluwer Academic Publishers.