Meaning in Context

Henning Christiansen[1] and Veronica Dahl[2]

[1] Roskilde University, Computer Science Dept.
P.O.Box 260, DK-4000 Roskilde, Denmark
E-mail: henning@ruc.dk

[2] Dept. of Computer Science
Simon Fraser University
Burnaby, B.C., Canada
E-mail: veronica@cs.sfu.ca

Abstract. A model for context-dependent natural language semantics is proposed and formalized in terms of possible worlds. The meaning of a sentence depends on context and at the same time affects that context representing the knowledge about the world collected from a discourse. The model fits well with a "flat" semantic representation as first proposed by Hobbs (1985), consisting basically of a conjunction of atomic predications in which all variables are existentially quantified with the widest possible scope; in our framework, this provides very concise semantic terms as compared with other representations. There is a natural correspondence between the possible worlds semantics and a constraint solver, and it is shown how such a semantics can be defined using the programming language of Constraint Handling Rules (Frühwirth, 1995). Discourse analysis is clearly a process of abduction in this framework, and it is shown that the mentioned constraint solvers serve as effective and efficient abductive engines for the purpose.

1 Introduction

Natural language semantics, or the relation between language and the world, has been one of the main concerns of linguists and computational linguists over the past few decades. Most efforts have gone into devising representation schemes for the world knowledge a sentence expresses. A plethora of such formalisms has resulted from different ways of viewing the world and how to represent it.

Compositional styles of semantics in the Montague tradition [17] typically result in interesting but very large and complex semantic expressions, involving functions, disjunctions, nested expressions, etc. It has recently been argued [18] that most approaches to semantics so far do not constitute semantics at all, but an alternative kind of syntax, that is: instead of being interpretations in the logical sense, they are simply elaborate paraphrases of the natural language utterance they purport to be the "meaning" of.

We propose instead a model of context-dependent natural language semantics in which sentence meanings are expressed in a context with which they interact

Proceedings of CONTEXT'05,
the Fourth International and Interdisciplinary Conference on Modeling and Using Context,
Paris, France, 5-8 July 2005.
Dey, A., Kokinov, B., Leake, D., Turner, R. (eds.),
Lecture Notes in Artificial Intelligence 3554. pp. 97-111

as opposed to being functions parameterized by context. Details of a sentence are interpreted in context and may in turn contribute to this context, so that the terms representing a particular sentence comes closer to a purified form of the "intention" of that sentence. Depending on application, we may even go so far as to say that the meaning of a sentence *is* its contribution to context.

This model is formulated in terms of possible worlds so that a context, here taken to mean the knowledge about the world collected so far, designates a class of worlds which are compatible with that knowledge. Our representations are true interpretations, in the sense that utterances analyze into a dynamically growing knowledge base which models the world so far, and provides as well the context within which future utterances will be analyzed. Individuals in the world do relate to names (constants in the knowledge base), relationships in the world to predicates in the knowledge base, and so on, as in any interpretation in the logical sense. Successive utterances weed out other possible worlds by further materializing the emerging world being described, so that there is a dynamic interaction between context and interpretation.

Example 1. Consider an everyday discourse that refers to different individuals, one of them usually called "Peter", and let us abstractly identify the individual as `p17`. In that discourse we take the meaning of "Peter" as a reference to this particular individual. In the same discourse, the occurrence of the pronoun "he" may refer to the same individual, and in our model we consider the meaning of this "he" to be the reference to `p17`. That occurrence of "he" could have been replaced by "Peter" without changing the overall meaning of the sentence or discourse, the pronoun is used for convenience only. (This as opposed to considering the meaning of "he" as an occurrence of a generic reference device radically different from a proper noun.)

Another speaker may refer to "the tall, red-haired man carrying a laptop"; the meaning in context of this expression may be exactly the same as the mentioned occurrences of "Peter" and "he", namely the reference to `p17`. The speaker may have chosen the longer expression for different reasons: he may not know the name, or may want to communicate to other people whom he (the speaker) believes do not know the name. In our Meaning in Context model, we see the meaning of the sentence (1) below as a wish to indicate (or: as having the purpose of indicating) that a particular individual has won a particular piece of hardware, and not as a wish to indicate other (and in a certain sense irrelevant properties of the individual) which abstractly might be represented by a logical fact `won(p17,f450)`.

The tall, red-haired man carrying a laptop won a brand new Ferrari. (1)

However, this sentence *presupposes* that the individual in question possesses certain properties such as being tall, etc. While the concise (contextual, or pragmatic) meaning of (1) may be represented by the expression `won(p17,f450)`, this can also be seen as just another contribution the meaning of the discourse.

From the viewpoint of discourse analysis, (1) provides the following little knowledge base,

```
tall(X), red_haired(X), carries(X,Y), laptop(Y),          (2)
        won(X,Z), ferrari(Z), brand_new(Z)
```

The use of absolute references such as `p17` and `f450` may be problematic from a philosophical point view and may also make the implementation difficult, so we have replaced them by logical variables.

In general, our methodology is applicable to "flat" meaning representations (e.g., those involving a conjunction of atomic predications in which all variables are existentially quantified with the widest possible scope). Of course, this leaves open the question of fine-tuned treatments of quantification. Previous approaches (e.g., [11]) maintain existential quantifiers only by tricks such as reifying universal variables as typical elements of sets and by a sort of skolemization of dependent existentially quantified variables. Future work would have to incorporate a subtler treatment that considers as well the many nuances in NL quantifiers. However, flattening everything into predications does have the advantage of allowing direct knowledge base creation — that is, direct construction of a true interpretation — as a result of an utterance's analysis, and one that takes context into account at that.

Discourse analysis based on this Meaning-in-Context model related to interpretation by abduction [12] as it consists basically of "guessing" those real-world hypotheses that are necessary for the discourse to reflect the world in a truthful way. The possible worlds semantics underlying our approach fits particularly well with recently developed techniques for language analysis and abduction based on the paradigm of Constraint Handling Rules [10] which is a declarative extension to the Prolog programming language for writing constraint solvers; we can, in fact, demonstrate a natural correspondence between the possible worlds semantics and such constraint solvers. We refer to implementations using two paradigms, CHR Grammars [4, 6] which provide a bottom-up analysis system in which parsing itself is treated as a constraint solving process and A^2LP [7] which incorporates abduction and other facilities in a more traditional Prolog and DCG setting. These technologies can handle contexts consisting of a variety of hypotheses: atomic hypotheses as shown above, so-called explicit negation of hypotheses, assumptions in the sense of Assumption Grammars [9], and also implications with existentially (globally) as well as universally (locally) quantified variables.

We can mention other advantages of the approach that are rather obvious, but not shown in details here due to lack of space. Lexical ambiguities that are difficult to resolve otherwise can be handled be taking the current context into account, or perhaps being delayed until more context information have been collected in the subsequent discourse. We can also provide rules that activate pre-existing contexts when sufficient amount of indication is found, thus making available new vocabulary and ontology.

The present paper is inspired by earlier work presented at CONTEXT'99 [3] but here given simpler and more general form and accompanied by relevant implementation techniques developed in the meantime.

2 General presentation of the Meaning in Context model

A discourse is grounded on some kind of real world. We use *real world* as a metaphor for some imagined or historically placed setting, in which events can take place and different properties may or may not hold, perhaps limited in time and space. It may, for example, be the planet Earth in the 20th century or the setting for an adventure which includes true unicorns and witches who really fly by means of brooms. A professionally constructed set of lies also reflects a "real world" in this sense, i.e., the illusion of a reality that someone wants to convince the public exists.[3] Real worlds are unwieldy entities that cannot be characterized fully by any set of facts, be it finite or infinite.

The word "context" is used by different authors and communities for different but often interrelated and dependent notions. Linguists often refer to the context of a phrase or word as the text that surrounds it. Another everyday usage of "context" refers to a section of the real world in which some events or a discourse takes place, and is often intertwined and confused with another meaning, namely knowledge about the same thing.[4] Our usage complies with the last view: to us, a *context* is a collected body of represented knowledge, typically what has been learned by attending a given discourse up to a certain point. In this respect, we agree with Stalnaker's work, and the possible worlds, Kripke-style semantics for contexts and the terminology we introduce is essentially a formalization of his ideas summarized in [21] .

Let $\mathcal{W}$ denote the set of all worlds, one of which may be designated the *real* world (the existence of a particular real world is, however, not essential for our purpose). No specific representation of context needs to be fixed, but we will, however, assume that it is decomposable in the sense that it can be separated into statements or propositions that we call *context facts* whose truth can be evaluated separately. Let, thus, $\mathcal{C}$ refer the set of all context or parts thereof and assume an operation $\cup : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{C}$ being associative, commutative, and idempotent; intuitively, no distinction is made between a proposition and the set consisting of it.

The notion of truth is represented by an entailment relation $\models: \mathcal{W} \times \mathcal{C}$ with the intuitive meaning that $w \models c$ if and only if c is a correct property in w. Entailment is *monotonic* in the sense that

$$w \models c_1 \cup c_2 \quad \text{implies that} \quad w \models c_1 \;\text{ and }\; w \models c_2.$$

This property should not be confused with the distinction between monotonic and non-monotonic reasoning as our model by no means excludes non-monotonic logics for reasoning with context.

[3] In the present paper, we disregard the consequences of possible disagreement between such worlds and the real, real world.

[4] The definition (one out of two) given by Webster's dictionary [22] indicates this duality "...**2.** the set of circumstances or facts that surround a particular event, situation, etc."

We define the *semantic function* W which maps any element of $\mathcal{C}$ into a set of its possible worlds, i.e.,

$$W(c) =_{\text{def}} \{w \in \mathcal{W} \mid w \models c\}.$$

The following monotonicity property follows from the definition:

$$W(c_1 \cup c_2) \subseteq W(c_1) \cap W(c_2)$$

Whenever, for contexts c and c', we have that $W(c) \subseteq W(c')$ we say that c *logically implies* c', written $c \models c'$. In case $c \models c'$ and $c' \models c$, we say that the two contexts c and c' are *equivalent* which is indicated symbolically $c \equiv c'$.

In general, it may be expected that $W(c)$ is either infinite or empty; a context c is *inconsistent* whenever $W(c) = \emptyset$ and *consistent* otherwise. Let $\top$ denote a prototypical inconsistent context and $\perp$ a prototypical context without information content at all, i.e., with $W(\perp) = \mathcal{W}$; a context with one of these properties is identified with the relevant of $\top$ and $\perp$.[5]

The notion of a discourse is not restricted particularly to text or speech, and applies also to general sequences of sensor signals or other sorts of meaning-bearing events, or a perhaps a mixture of all kinds.[6]

An alphabet Σ is assumed, and we define a *sentence* as an element of some designated subset $\mathcal{S}$ of Σ^+.[7] A *discourse* is any finite concatenation of sentences, i.e., we let

$$\mathcal{D} =_{\text{def}} \{s_1 \cdot \ldots \cdot s_n \mid n \geq 0, s_i \in \mathcal{S}\}$$

refer to the set of all discourses; ϵ denotes the empty discourse, and concatenation is indicated by juxtaposition or made explicit (as above) by a dot. For simplicity of definition, it is assumed that the decomposition of a discourse into sentences is unique, but otherwise a discourse may be as ambiguous as ever.

A *(contextual) meaning function* is a function $M : \mathcal{D} \rightarrow \mathcal{C}$ which satisfies the following *prefix properties*:

$$\begin{aligned} M(\epsilon) &= \perp \\ M(d \cdot s) &\models M(d), \text{ for any discourse } d \text{ and sentence } s \end{aligned}$$

In other words, a meaning function M extracts the content of a discourse and puts in into a represented form, and the composition of M and W provides a possible worlds semantics for discourses. The second prefix property indicates that the longer discourse, the more context knowledge is learned and the fewer worlds possible. We may classify as *rubbish* (*boring*), any discourse d with $M(d) = \top$ $(= \perp)$.

[5] The symbols $\top$ and $\perp$ correspond to top and bottom elements in the algebraic lattice of $\mathcal{C}$ with $\cup$ induced by W.

[6] Whether or not a discourse is considered an element in the real world is not important for us at this level; this may or may not be assumed as convenient.

[7] The term "sentence" should be taken as a convenient usage for any syntactic entity that directly contributes to context and do not necessarily comply directly with a syntactic notion of a sentence.

We assume also the following about M, which we may call *syntax independence.* For any discourses d, d', and sentence s we have that

$$\text{if } M(d) \equiv M(d'), \text{ then for any sentence } s, \text{ we have } M(d \cdot s) \equiv M(d' \cdot s)$$

This means that the interpretation of a sentence depends on what has been said and not the way it has been said. This property indicates that any piece of information that can affect the interpretation of future sentences needs to be passed through the context, so one way to handle coordination of parallel sentences is to include (the memory of) the already spoken words as part of the context.

Finally, we introduce the notion of the *accomodation function* $A : \mathcal{C} \times \mathcal{S} \to \mathcal{C}$ given by a meaning function M which is intended as a formal equivalent to Stalnaker's notion of accomodation [21]:

$$A(M(d), s) = M(d \cdot s), \text{ for any discourse } d \text{ and sentence } s$$

We notice that a meaning function gives rise to an accommodation function and vice versa.

Example 2. The very first sentence in Nikos Kazantzakis' novel about Zorba the Greek [15] consists of the following four words in Greek.

Τον	*πρωτογνώρισα*	*στον*	*Πειραία*	
him	first-meet/know-1stSingPast	in-the	Pireus	(3)

Greek grammar is very compact, for example the -α inflection of the verb indicates first person, singular, and past tense, so the subject is implicit in the sentence. It translates into "I met him for the first time in Pireus". The author has a specific real world in mind, and even the opening sentence (3) of the novel provides us quite a lot of important information about this world, and actually information that is central to the very end of the story. In the terms defined above, an accomodation function may be involved which produces a first context telling that at least two persons are involved in the story, one of whom is the novel's "I", and something about the relationship between the two, etc. In a formal, flat representation this context might be something like this:

```
person(X), male(X), person(Y), i(Y), X≠Y, event(E),
     past(E), place_of(E,P), name_of(P,Pireus),          (4)
meeting_event(E,X,Y), knows(X,Y), first_of_its_kind(E)
```

A meaning function assigns a set of possible worlds to this context, worlds that all includes Pireus and two persons who made each others acquaintance there, etc. As the novel's discourse goes on we learn more and more about `X` and `Y` and other events in which they and other characters are involved, thus narrowing down to fewer and fewer worlds. For the indicated representation and its meaning function, it may be assumed that any context is inconsistent if it contains `event(E1)`, `event(E1)`, `before(E1,E2)`, `first_of_its_kind(E2)`.

3 Discourse analysis as constraint solving

The analysis of a given discourse will be considered in a sequential manner, reflecting the time span in which the discourse takes place and the fact that a discourse may continue even if we thought that it was finished.

In this section we refer to constraints in the sense of constraint logic programming (CLP) [10, 13]. A *constraint* is a formula of (typically) first-order logic, often restricted to atomic formulas, but this needs no be the case; a *constraint store* is a finite set of constraints. A semantics is assumed for constraints and constraint stores, e.g., in terms of logical axioms.

A *normal form* is assumed among semantically equivalent constraint stores, which can be briefly characterized as having a minimal textually representation that is an acceptable answer to a user. We may as an example expect that $\{p(x), q(y), x = y\}$ has the normal form $\{p(x), q(x)\}$ (in case p and q do not depends on each other and the equality sign has the usual meaning). Special normal forms are *false* (failure) and *true* (the empty constraint store) which, in our applications to discourse analysis are equivalent to $\top$, resp., $\bot$.

A *constraint solver* keeps the constraint store normalized in an incremental fashion, i.e., when a new constraint arrives to a normalized store, it re-establishes normalization while respecting semantics. In practice, however, a constraint solver often only approximates this property by, for example, returning a store that is equivalent to *false* but given as a set of other constraints. This may be due to inherent, undecidable properties or simply as a shortcut made for reasons of efficiency. For ease of definitions, we ignore the possible imperfections of practical constraint solvers and accept *the* stores produced by the constraints solvers as "normalized".

A CLP system consists of a *driver* that produces constraints during the process of solving some problem, and the constraint solver maintains normalization. Typically, the driver is a Prolog engine running some program and the constraint solver reports back successes, failures, and implied unifications.

For discourse analysis, we identify context with constraint store and, thus, context facts with constraints. A constraint solver is thus, a function $C : \mathcal{C}_{\text{norm}} \times \mathcal{C} \to \mathcal{C}_{\text{norm}}$ where $\mathcal{C}_{\text{norm}}$ refers to the set of constraint stores (contexts) in normal form. The fact that a constraint solver must respect the semantics of the constraints can now be expressed as $C(c, c') \equiv c \cup c'$.

A driver for discourse analysis can be a parser written in Prolog or any other kind of syntax analyzer working in a sequential manner. A syntax analyzer can be depicted as a function S from sentences to constraints; in general the analyzer may utilize the current constraint store so we can write $S : \mathcal{S} \times \mathcal{C}_{\text{norm}} \to \mathcal{C}$.[8] So when talking about syntax analysis, we have abstracted away the possible construction of a syntax tree or similar representation, and indicate only the "semantic tokens" produced in the shape new constraints (i.e., new context facts).

[8] In many cases, it is sufficient with a context independent analyzer $S : \mathcal{S} \to \mathcal{C}$ as the constraint solver will adapt the constraints produced to the current context.

A syntax analyzer S is correct with respect to accomodation function A whenever

$$A(c,s) = C(c, S(c,s)) \text{ for any normalized context } c \text{ and sentence } s.$$

The other way around, if S is the given entity, we can say that it defines an accommodation function and in turn a meaning function by the condition above.

Disjunctions and context splitting

It may be the case that new context facts indicate a choice between two or more incompatible hypotheses, for example, that the pronoun "he" refers to either Peter or Paul, but with no indication of which is the right one. We can assume the context c_0 being extended with the new piece of information $x = \mathit{peter} \lor x = \mathit{paul}$.

In principle, a constraint solver could treat disjunctions as constraints, i.e., consider the formula $x = \mathit{peter} \lor x = \mathit{paul}$ as *one* constraint. This does not fit with standard CLP technology that eliminates equations by means of unification, i.e., replacing any occurrence of a variable by the indicated value. Instead the constraint store is made subject of a *splitting*, which means that constraint store

$$c_0 \cup \{x = \mathit{peter} \lor x = \mathit{paul}\}$$

gives rise to two new constraint stores

$$c_0 \cup \{x = \mathit{peter}\} \quad \text{and} \quad c_0 \cup \{x = \mathit{paul}\}$$

and the analysis of the remainder of the discourse needs to be performed twice, once for each store and, of course, recursively in case the store splits again later.

This notion of splitting in a constraint solver has been formalized in the language of CHR$^\lor$ [2]. A CLP system based on Prolog implements splitting by means of backtracking. Another strategy is to maintain the different constraint stores in parallel, so that when a new context fact comes in from the discourse, a copy is added to all different stores each of which is normalized before next sentence is analyzed; this is applied in the CHR Grammar system [6] which is briefly mentioned below. We leave out the straightforward formalization in terms of possible worlds of systems including splitting (based on the union of possible worlds for the alternative contexts).

Constraint Handling Rules

The language of Constraint Handling Rules, CHR, is an extension to Prolog intended as a declarative language for writing constraint solvers for CLP systems; here we give a very compact introduction and refer to [10] for details.

Constraints are first-order atoms whose predicate are designated constraint predicates and a constraint store is a set of such constraints, possible including variables, that are understood existentially quantified at the outermost level. A constraint solver is defined in terms of rules which can be of the following two kinds.

$$\text{Simplification rules: } c_1, \ldots c_n \texttt{ <=> } \mathit{Guard} \mid c_{n+1}, \ldots, c_m$$
$$\text{Propagation rules: } c_1, \ldots c_n \texttt{ ==> } \mathit{Guard} \mid c_{n+1}, \ldots, c_m$$

The c's are atoms that represent constraints, possible with variables, and a simplification rule works by replacing in the constraint store, a possible set of constraints that matches the pattern given by the *head* $c_1, \dots c_n$ by those corresponding constraints given by the *body* $c_{n+1}, \dots, c_m$, however only if the condition given by *Guard* holds. A propagation rule executes in a similar way but without removing the head constraints from the store. In addition, rule bodies and guards may include equalities and other standard relations having their usual meaning. The declarative semantics is hinted by the applied arrow symbols (bi-implication, resp., implication formulas, with variables assumed to be universally quantified) and it can be shown that the indicated procedural semantics agrees with this. This is CHR explained in a nutshell.

The declarative semantics provides a possible worlds semantics for constraint stores (alias contexts). To see this, define for a set of rules R, a *model* of R as any set of ground (i.e., variable-free) constraints with the property that any rule in R evaluates to true in M in the usual way (see, e.g., [10] for precise definitions or use any standard textbook of mathematical logic). In general, there may be an infinity of models of a set of rules R, which can be understood as a collection of all possible worlds that we refer to as $\mathcal{W}_R$. In general, we may use $\mathcal{W}_S$ to refer to the set of models of any formula or set of formulas S.

A constraint store c represents the knowledge collected up to a certain point and its possible worlds should be exactly all those models of R in which the information of c is true, i.e., those models of c that are also models of R. We define, thus,[9]

$$W_R(c) =_{\mathrm{def}} \mathcal{W}_R \cap \mathcal{W}_c = \mathcal{W}_{R \cup c}. \tag{5}$$

The condition (5) indicates also that the R can be thought of as defining an implicit initial context whose meaning function restricts to those worlds in which R holds.

Example 3. Documentaries assume implicitly the laws of physics whereas this is typically not the case for space movies.

As a consequence of the properties noticed earlier, we see that a CHR program together with a syntax analyzer induces an accommodation function and meaning function in the strict sense defined above.

Example 4. In example 2, we indicated a flat discourse representation which is well suited for CHR. A constraint solver for this representation may include the following rules.

```
X=/=X ==> fail.
before(E1,E2), first_of_its_kind(E1) ==> fail.
meeting_event(E,X,Y) ==> knows(X,Y).
place_of(E,X) \ place_of(E,Y) <=> X=Y.
i(X) \ i(Y) <=> X=Y.
```

[9] Strictly speaking, this does not comply with our assumption that a world cannot be characterized by any, even infinite, set of facts. It is easy to repair this flaw by the introduction of a "true" possible worlds semantics for logical models.

The first two rules identify inconsistent states, the next one adds a new hypothesis implied by another one. The rule about `place_of` tells that the place of an event is unique, and when it is applied during a discourse analysis, it may identify inconsistency if two different locations for the same event are asserted or help resolve an anaphoric reference. In the same way, the last rule indicates that the narrator “I” of a story is unique. Notice that the two last rules are so-called simpagation rules of CHR which indicate that the constraints following the backslash are removed from the constraint store and the others stay (thus they abbreviate special kinds of simplification rules).

In our own work, we have extended CHR so that the constraint store may contain dynamically created CHR rules as well, which is useful in many application for language processing. So, for example, the sentence *“all green objects are on the table”* can be modelled by adding `green(X) ==> on(X,the_table)` to the constraint store.

4 On the relation to abductive reasoning

For reasons of space, we cover this topic in a very compact manner; detailed arguments can be found in [4, 6, 7].

Generation of a discourse D is an inherently deductive process based on known premises of a grammar G and the speaker's known context C. The relationship between the components is that

$$G \wedge C \rightarrow D. \tag{6}$$

For this discussion, let us instantiate this pattern by assuming that G is a DCG [19], C a set of Prolog facts, and D an answer produced by a Prolog interpreter.

For discourse *analysis*, grammar and discourse are known but the premise C of (6) is the unknown to be found. This is by definition an abductive problem for which methods developed for Abductive Logic Programming [14] (ALP) apply. In general, a set of *integrity constraints* (ICs) are needed: ICs are logical conditions that must be satisfied by any context C for it to be consistent in the sense defined above. An ALP interpreter enforces the ICs and, in case of splitting, discards inconsistent branches. (In the case of generation, the ICs are not necessary, but they are implicit in the tacit assumption that the *given* C represents some real world.)

It has been explained above how a CHR program P defines a meaning function which especially can identify inconsistent contexts. Now comes an important point: When such a P is combined with a syntax analyzer written in Prolog (as a DCG, for example), the overall functioning is exactly that of an abductive interpreter with P as integrity constraints. When a discourse D is given as query, the corresponding context is produced as the final constraint store or, if splitting occurs, as the disjunction of the alternative constraint store produced.

5 Discourse analysis in A^{2}LP and CHRG

A^{2}LP is an extension to Prolog with abduction and assumptions defined by means of a few CHR rules explained in [7]. Such assumptions, known from Assumption Grammars [9], are related to abduction but provide scoping principles that are useful for modeling many linguistic phenomena; here we show only the abduction part of A^{2}LP. Prolog's built-in grammar notation can be used with A^{2}LP, and we show a grammar for simple discourses on still-life pictures.

Context representation is given as facts about the immediate physical relationship between objects, so, e.g., `i_on(`a`,`b`)` denotes that a is situated directly upon b, similarly for the predicate `i_in`. A constraint solver defines a meaning function or, alternatively in ALP terminology, provides the integrity constraints of an abductive logic program.

```
i_on(X,Y), i_on(Y,X) ==> fail.
i_in(X,X) ==> fail.
container(C) ==> thing(C).
i_in(the_box,the_vase) ==> fail.
i_in(_,C) ==> container(C).
```

It identifies a number of impossible situations and indicates properties and classes of some known objects, for example that `the_box` cannot be inside the `the_vase`. The following rules apply semicolon as in Prolog, i.e., disjunction by splitting, to restrict to a universe with exactly four objects.

```
thing(X) ==> X=the_flower ; X=the_box ; X=the_vase ; X=the_table.
container(X) ==> X=the_box ; X=the_vase.
container(the_flower) ==> fail.  container(the_table) ==> fail.
```

Finally, let us introduce a rule that defines the everyday notion of one thing being on another thing, which may or may not involve an intermediate object.

```
on(X,Y) ==> i_on(X,Y) ; i_on(X,Z), i_on(Z,Y) ; i_in(X,Z), i_on(Z,Y)
```

The following grammar rule defines a little syntax analyzer for simple sentences.

```
sentence --> [A,is,on,B], {thing(A), thing(B), on(A,B)}.
```

Analyzing *"the flower is on the table"* gives rise to a context constructed as the disjunction of different final states, which allows the placement of the flower in a number of different positions. Continuing the discourse with *"the flower is in the vase"* and assuming an analogous rule for sentences about *"in"*, we cut down to possible worlds in which the flower is placed in a vase on accordance with the traditional picture of the ideal home.

There is no reason to include more sophisticated examples as it is well-known that Prolog's DCGs can model a large variety of linguistic phenomena, and it is well-known that a flat representation in the sense of [11] is very general. In addition, the newest version of A^{2}LP also can handle of context facts in the shape of rules, as shown in the end of section 3.

It also clear that a constraint solver written in CHR can work together with other and more sophisticated syntax analyzer. Due to lack of space, we leave out a presentation of the CHR Grammar system [5, 6], but it must be mentioned as it provides a very flexible kind of grammars that are compiled into CHR rules that run as a robust bottom-up parser that can interact with abduction and constraint solvers written in CHR as shown above.

6 Extensions and application

We have only showed very simple examples but it should be emphasized that CHR is a very powerful language for expressing different properties; it is Turing-complete so every computable function can be used as meaning function. We indicate a few relevant applications.

Resolving lexical ambiguity. A classical example is "bank" which can be a river bank or a financial institution, but in most cases it is "clear from context" for a human what is meant. The following two rules provides a self-explanatory example of how this can be expressed.

```
nautical_event(E,X,Y) \ place(E,bank) <=> place(E,river_bank).
financial_event(E,X,Y) \ place(E,bank) <=> place(E,financial_bank).
```

They give a dynamic interaction between context and analysis. In case a relevant `nautical_event` is known at the point when `place(···,bank)` arrives, the rule applies immediately; otherwise the resolution of `place(···,bank)` may be delayed while analysis continues and done when the relevant context indicators are encountered.

Activation of predefined context elements. "... tire ... gearbox ... brake ... 200 kmh ...". There is no doubt these people are talking about cars, so "boot" is likely not a piece of footwear but a place for your luggage. A systematic way of treating this phenomenon is to write rules that dynamically sum up a number that reflects the number of indicators, and when this is sufficiently high, call a constraint that activates the context.

```
indication(carContext,K) ==> K>0.8 | activate_context(cars).
```

Here `activate_context` can be a premise in other rules (as above) but also be a predicate programmed in Prolog that installs a sub-lexicon.

Non-monotonic reasoning. The monotonicity assumptions in our general definitions does not exclude nonmonotonic reasoning as would be needed to handle liars in a more clever way that just noticing inconsistency. Here standard reification technique applies: add an extra attribute to each predicate in question, intitially uninstantiated, and set to `notTrusted` if it is learned that that the speaker is a liar. Intuitively, this may seem better than a traditional non-monotonic system that would *remove* the problematic facts: The human listener would still remember that these lies were told by a certain speaker and may have this in mind (i.e., in context) when analyzing what follows.

7 Related work

Seeing the meaning of linguistic expressions as context-dependent is not a new idea, and the principle has been applied in papers which are too numerous to mention here, including at the recent Context conferences. A precursor from 1975 is [20], presenting a context-dependent semantic model for natural language whose definition of interpretation function looks very much, at the surface, like our accommodation functions, but the remaining part of that model is difficult to compare to ours (which clearly has benefitted from insight gained in logic programming, especially variants with abduction and constraints).

A search on the web based on the title of the present paper gave an interesting hit, a paper entitled "Meaning in Context: Is there any other Kind?" [16] from 1979 that argues for a shift in attitude in phenomenology, sociolinguistics, and ethnomethodology based on very much the same intuition that we have based our work on.

The view of discourse analysis as abduction appears in many works in the last twenty years or more, with [12] usually considered a central reference; [4] gives more references. These methods have never become widespread in practice, and we may hope that the application of constraint logic programming techniques as we advocate may result in, or inspire to, efficient and useful systems.

Abduction in CHR was proposed by [1] and refined in other publications already mentioned; basically it can be explained as a transformation of abduction into deduction, and it is interesting to compare this with a paper from 1991 [8] (at a time when CHR did not exist) that pointed out an isorphism between a class of abductive problems and deduction.

8 Conclusion and perspectives

We have argued for a Meaning-in-Context approach to natural language semantics, grounded on overall conceptual considerations, given a possible worlds formalization of it, and in a straightforward way explained effective implementations based on recent advances in constraint logic programming. CHR is the subject of intensive research and development these years, including on adding priorities and weightings to the language — which will be an essential addition for the paradigm presented here.

Acknowledgements This work is supported by the CONTROL project, funded by Danish Natural Science Research Council, the Velux Visiting Professor Programme, and Canada's NSERC Discovery Grant program.

References

1. Abdennadher, S., and Christiansen, H., An Experimental CLP Platform for Integrity Constraints and Abduction. Proceedings of FQAS2000, Flexible Query Answering Systems. Larsen, H.L., Kacprzyk, J., Zadrozny, S., (Eds.) *Advances in Soft Computing series,* Physica-Verlag (Springer), pp. 141–152, 2000.
2. Abdennadher, S., and Schütz, H. CHR$^{\vee}$: A flexible query language. Proceedings of FQAS'98, Flexible Query Answering Systems. Andreasen, T., Christiansen, H., Larsen, H.L. (Eds.) *Lecture Notes in Artificial Intelligence* 1495, pp. 1–14, Springer, 1998.
3. Christiansen, H., Open theories and abduction for context and accommodation. 2nd International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT'99) Bouquet, P., Brezillon, P., Serafini, L. (eds.) *Lecture Notes in Artificial Intelligence* 1688. Springer-Verlag, pp. 455–458, 1999.

4. Christiansen, H., Abductive Language Interpretation as Bottom-up Deduction. In: Natural Language Understanding and Logic Programming, Proceedings of the 2002 workshop, ed. Wintner, S., *Datalogiske Skrifter* vol. 92, Roskilde University, Comp. Sci. Dept., pp. 33–47, 2002.
5. Christiansen, H., CHR Grammars web site. Source text, manual, and examples, http://www.ruc.dk/˜henning/chrg, 2002.
6. Christiansen, H., CHR grammars. *International Journal on Theory and Practice of Logic Programming, special issue on Constraint Handling Rules*, to appear 2005.
7. Christiansen, H., Dahl, V., Assumptions and Abduction in Prolog *Proceedings of WLPE 2004: 14th Workshop on Logic Programming Environments and Multi-CPL 2004: Third Workshop on Multiparadigm Constraint Programming Languages Workshop Proceedings*, September 2004, Saint-Malo, France. Susana Muñoz-Hernández, José Manuel Gómez-Perez, and Petra Hofstedt (Eds.) pp. 87-101.
8. Console, L., Theseider Dupré, D., Torasso, P., On the Relationship between Abduction and Deduction. *Journal of Logic and Computation* 1(5), pp. 661–690, 1991.
9. Dahl, V., and Tarau, P. From Assumptions to Meaning. *Canadian Artificial Intelligence* 42, Spring 1998.
10. Frühwirth, T.W., Theory and Practice of Constraint Handling Rules, *Journal of Logic Programming*, Vol. 37(1–3), pp. 95–138, 1998.
11. Hobbs, J., Ontological Promiscuity. *Proceedings of the 23rd conference on Association for Computational Linguistics*, 8-12 July 1985, University of Chicago, Chicago, Illinois, USA, Proceedings, ACL, pp. 61–69, 1985.
12. Hobbs, J.R., Stickel, M.E., Appelt D.E., and Martin, P., Interpretation as abduction. *Artificial Intelligence* 63, pp. 69-142, 1993.
13. Jaffar, J., Maher, M.J., Constraint logic programming: A survey. *Journal of logic programming*, vol. 19,20, pp. 503–581, 1994.
14. Kakas, A.C., Kowalski, R.A., and Toni, F. The role of abduction in logic programming, *Handbook of Logic in Artificial Intelligence and Logic Programming*, vol. 5, Gabbay, D.M, Hogger, C.J., Robinson, J.A., (eds.), Oxford University Press, pp. 235–324, 1998.
15. Kazantzakis, N., *Βίος και Πολιτεία του Αλέξη Ζορμπά.* [Eng.: Life and career of Alexis Zorbas.] 1946.
16. Mishler, E.G., Meaning in Context: Is there any other Kind? *Harward Educational Review* Vol. 49, No. 1, pp. 1–19, 1979.
17. Montague, R., *Formal philosophy* Yale University Press: New Haven.
18. Penn, J.R., The Other Syntax: Approaching Natural Language Semantics through Logical Form Composition To appear in "Proceedings of first International Workshop on Language Processing and Constraint Solving, Roskilde, Sept. 1–3, 2004", eds. Christiansen, H., Skadhauge, P.R., Villadsen, J., *Lecture Notes in Artificial Intelligence* 3834, 2005.
19. Pereira, F.C.N., and Warren, D.H.D., Definite clause grammars for language analysis. A survey of the formalism and a comparison with augmented transition grammars. *Artificial Intelligence* 10, no. 3–4, pp. 165–176, 1980.
20. Rieger, C., Conceptual overlays: A mechanism for the interpretation of sentence meaning in context. *Proc. IJCAI-75* pp. 143–150.
21. Stalnaker, R., On the representation of context. *Journal of Logic, Language, and Information*, vol. 7, pp. 3–19, 1998.
22. *Webster's Encyclopedic Unabridged Dictionary of the English Language.* 1989 Edition, Gramercy Books, dilithium Press, 1989.