DRAFT

Frame Theory, Dependence Logic and Strategies

Ralf Naumann and Wiebke Petersen

Institut für Sprache und Information
Universität Düsseldorf
Germany

Abstract. We present a formalization of the Löbner-Barsalou frame theory (LBFT) in Dependence Logic with explicit strategies. In its present formalization, [Pet07], frames are defined as a particular kind of typed feature structures. On this approach, the semantic value of a lexical item is reduced to its contribution to the truth conditions of sentences in which it occurs. This reduction does neither account for dynamic phenomena nor for results from neuroscience which show that meaning cannot be reduced to truth conditions. In order to overcome these shortcomings, we develop a dynamic frame theory which is based both on Dependence Logic [Vää07] and Dynamic Epistemic Logic ([vB11]). The semantic phenomenon with respect to which this framework is tested are numerical expressions like 'two' or 'at least two'. They are interpreted as strategies which change the input information state to which they are applied.[1]

Keywords: dependence logic, dynamic epistemic logic, dynamic semantics, numerals, scalar quantifiers

1 Introduction

In the last two decades, there has been a growing interest in combining ideas from cognitive science, neuroscience (neurophysiology and neuroimaging), formal linguistics and computer science, using advanced tools from mathematical logic. A major reason for this trend can be seen in improved empirical methods of testing linguistic theories with respect to their empirical and cognitive adequacy: (i) How are sentences (or expressions occurring in them) processed in the brain? and (ii) How is semantic knowledge about the meanings of words acquired in the process of language learning? In this paper we will focus on one particular linguistic phenomenon: bare numerals like 'two' or 'three' and so-called scalar expressions like 'at least', 'at most', 'more than', 'less than', and 'exactly', which can be used to modify bare numerals.

(1) a. John read (at least / at most / exactly) two books.
b. Mary drank (less than /more than) three cups of coffee.

[1] The research was supported by the German Science Foundation (DFG) funding the Collaborative Research Center 991.

There are at least two reasons for hoosing this particular topic. First, it has received much attention in recent years, not only from linguists but also from cognitive and neuroscientists. Second, and even more importantly, the theoretical analyses and empirical results presented in those studies provide ample evidence for the fact that these expressions cannot be analyzed in such a way that their meaning is reduced to the contribution they make to the truth conditions of sentences in which they occur.

There is no general agreement what exactly this additional, non-truth-functional meaning component should be. We will show that in order to answer this question one has to take up ideas from different conceptual and theoretical frameworks. In particular, we will argue for the following interdependent theses: (i) In principle, there is no difference between linguistic and non-linguistic meaning (Situation Theory), (ii) The meaning of a linguistic expressions is its *context change potential* (Dynamic Semantics, Update Semantics); (iii) The meanings of linguistic expressions are closely related to how the information (or belief) state of an agent changes when processing an utterance containing this expression (Dynamic Epistemic Logic) and (iv) The meanings of linguistic expressions are closely related to the notions of a strategy from game theory and that of a plan from cognitive science and philosophy.

2 Bare numerals and scalar quantifiers

2.1 Bare numerals

There are two different semantic analyses of bare numerals like 'two'. According to the first one, the set-theoretical condition imposed by 'two' is the same as that for 'at least two' (2). This is, following [Hor89], the so-called 'one-sided' analysis.

(2) $[[two]] = \{\langle P, Q\rangle : |P \cap Q| \geq 2\}$

On this analysis, bare numerals form a scale such that the following inferences are true with respect to this scale.

(3) a. Joe has four children → John has three {two, one} children.
b. John doesn't have three hildren. → John doesn't have {five, six, . . . } children.

The 'exactly' interpretation arises if an implicature with respect to such a scale is used. Evidence for this semantic analysis comes from the fact that such an implicature can be canceled.

(4) a. Pat has three children and possibly four.
b. Pat has three or even four children.
c. Pat doesn't have three children. → Pat has less than three children.

By contrast, on a 'two-sided' analysis, the set-theoretic condition is $\{\langle P, Q\rangle : |P \cap Q| = 2\}$. If the meaning of an expression is reduced to its truth-conditions

(or its contribution to the truth-conditions of sentences), 'two' and 'at least two' have the same meaning on a one-sided analysis. By contrast, given the same assumption, 'two' should be equivalent to 'exactly two' on a two-sided analysis. Both analyses face empirical problems. The most important of these problems is that bare numerals give rise to different interpretations, depending on the context in which they occur ([Mus04], [Car98], [Sza10, 145ff.]).

(5) A: How many mistakes did you make?
B: I made three mistakes.

The preferred interpretation of 'three' in (5) is 'exactly three'. Similarly, on a predicative or a collective use, a bare numeral gets an 'exactly'-reading.

(6) Those are two dogs.
(false if the speaker is pointing at three dogs)

(7) Two dogs (together) pulled the sled to the barn.
(false if the collective agent of the event consisted of three dogs, or of two dogs and a sheep)

By contrast, if a bare numeral is used distributively, the preferred interpretation of 'three' is 'at least three' (8). E.g., (8a) is true even if there are more than two dogs which barked.

(8) a. Two dogs were hungry. They barked.
b. You need to make three mistakes to be allowed to take the test again.

Finally, a bare numeral can also receive an 'at most' interpretation, witness (9), the preferred interpretation of which is that the addressee passes the test if (s)he makes *at most* three mistakes.

(9) You can make three mistakes and still pass the test.

2.2 Scalar quantifiers

If the meaning of a (modified) generalized quantifier is defined solely in terms of its truth conditions, which, in turn, are defined purely set-theoretically, the pairs in (10) and (11) are semantically equivalent.

(10) a. John read at least three books.
b. John read more than two books.

(11) a. John read at most three books.
b. John read fewer than four books.

This view of defining the meaning of scalar quantifiers has been criticized both for empirical reasons and from the perspective of language acquisition.

Language acquisition [Mus04] conducted different experiments with 5-year old, preschooler children in order to assess their semantic competence with respect to bare numerals like 'two' and modified bare numerals like 'at least two' or 'exactly two'. The aim of the first two experiments was to access the ability of those children to differentiate between an 'exactly' and a non-'exactly' interpretation of bare numerals. The findings are given in Table 1 (percentage indicates the acceptance rates).

context / group	**exactly n**	**at least n**	**at most n**
adult	100%	95%	97,5%
child	100%	80%	83,5%

Table 1. Results of an experiment on comparing the availability of readings for bare numerals for children aged five [Mus04]

The main finding of this experiment is that preschoolers aged 5 can assign numerals the full range of interpretations available in the adult grammar (i.e. 'exactly n', 'at least n' and 'at most n') ([Mus04, 30]). The second experiment aimed at comparing the children's semantic knowledge of bare numerals that are modified with 'exactly', 'at least', 'at most' or 'more than', Table 2.

context / group	**exactly n**	**at least n**	**at most n**	**more than n**
adult	100%	100%	95,5%	-
child	100%	50%	54,1%	88%

Table 2. Results of an experiment on comparing the semantic knowledge of children aged five for modified bare numerals [Mus04]

[Mus04] draws the following consequences from this experiment: (i) Children aged 5 know what 'exactly n' and 'more than n' mean but they are clueless about the meaning of phrases like 'at least n' and 'at most n'; (ii) Children do not disregard the modifier expression, witness the high acceptance rate for the comparative 'more than' and (iii) Although children have implicit knowledge of the fact that bare numerals can have exact and non-exact interpretations, they do not yet know the meaning of the expressions corresponding to the non-exact interpretations of bare numerals.

[GKC+10] tested the semantic knowledge of 11-year-old children with respect to modified bare numerals, also including 'fewer than', which was absent from the study carried out in [Mus04]. The results in descending order indicate the percentage of correct answers: (i) 'exactly' (100 %), (ii) 'more than' (97%), (iii) 'at least' (88%), (iv) 'fewer than' (77%), 'at most' (43 %). This experiment shows that in contrast to 5-year-old children, 11-year-old children do very well with 'at

least' (88%), but they still have significant difficulties in understanding 'at most' (43%). [GKC+10, 143] comment: 'While five-year-olds have serious trouble with superlative quantifiers, they are quite good with 'more than'. By the time they are 11, children are essentially perfect with 'more than', still struggling with 'at most', and fairly good with 'at least' and 'fewer than'. [PM03] showed 5-year-old children a scenario in which (exactly) three horses jumped over a fence. At the end of the story, a puppet described what happened by uttering (12a).

(12) a. Two of the horses jumped over the fence.
b. Exactly three horses jumped over the fence.

The comment (12a) of the puppet was consistently rejected by the hildren. They argued that *three* and not just *two* horses jumped over the fence, i.e. 'two' is not interpreted as 'at least two'. The result of this experiment therefore shows that given a particular scenario, children are able to determine a unique interpretation for a bare numeral.

The 'namely'-construction In contrast to comparative scalar modifiers superlative scalar modifiers followed by the 'namely'-construction allow a specific or referential reading, conveying the information that the speaker has a particular set of persons in mind, (13) ([GN07, 534]).

(13) a. I will invite at least two people, namely Jack and Jill.
b. ?I will invite more than one person, namely Jack and Jill.
c. *I will invite more / fewer than two people, namely Jack and Jill.

3 A frame-based analysis of bare numerals and scalar modifiers

3.1 Frames, feature structures and teams

The discussion of the data in section 2 has shown that bare numerals allow different interpretations and that veridical visual observations are interpreted in a unique way, leaving no way for epistemic uncertainty.[2] Using a dynamic framework, in which meanings are context change potentials, we can give the following possible analysis of the above data: (i) linguistic inputs like utterances or speech acts involving bare numerals update the information state of an agent in a non-deterministic way because (s)he cannot epistemically distinguish between the different interpretations receiving only this input, and (ii) for a veridical observation, the exact number can be exactly determined (provided it is not too large). To make this idea precise, consider the following example taken from [vB11, 45f.].

[2] Of course, this need in general not to be true for arbitrary observations. Here we refer to the circumstances under which the hildren and the adults observed the scene where three horses jumped over a fence.

> **Throwing a party:** You know that (i) John comes if Mary or Ann does, (b) Ann comes if Mary does not come, (c) If Ann comes, John does not.

The question is what information can be deduced from this set of premises? Using first-order reasoning, one gets:

> By (c), if Ann comes, John does not come. But by (a), if Ann comes, John comes. This is a contradiction, so Ann does not come. But then, by (b), Mary comes. So, by (a) once more, John must come. Indeed a party {John, Mary} satisfies all three premises.

As noted in [vB11, 46], the premises can equally be seen from a dynamic perspective on which they are taken as information events, like observations or utterances by others, which change the information state an agent is in. At the beginning, no information is available to the agent so that all eight options of inviting three different persons are possible.

(14) $\{MAJ, MA\neg J, M\neg AJ, M\neg A\neg J, \neg MAJ, \neg MA\neg J, \neg M\neg AJ, \neg M\neg A\neg J\}$

The three premises are formally taken as *updating* this initial information state the agent is in. A possible sequence of updates is given in (15).

(15) (M or A) $\rightarrow$ J, new state : $\{MAJ, M\neg AJ, \neg MAJ, \neg M\neg AJ, \neg M\neg A\neg J\}$
not-M $\rightarrow$ A, new state : $\{MAJ, M\neg AJ, \neg MAJ\}$
A $\rightarrow$ not-J, new state : $\{M\neg AJ\}$

Applying this type of reasoning to the sentence 'Three horses jumped over the fence, involving the bare numeral 'three', one arrives at the following sequence of deliberations. Only getting this information, the agent doesn't know whether 'three' has to be interpreted as 'at most three' (L), 'exactly three' (E) or 'at least three' (M).[3] Thus, at least theoretically, there are eight options, (16).

(16) $\{MEL, ME\neg L, M\neg EL, \neg MEL, M\neg E\neg L, \neg ME\neg L, \neg M\neg EL, \neg M\neg E\neg L\}$

Given that 5-year-old preschoolers can already distinguish between the three principle cases in the sense that they know that in a given context exactly one option is true, he or she applies the four rules in (17), reducing the eight options in (16) to the three given in (18).[4]

(17) a. $E \rightarrow \neg(M \vee L)$
b. $M \rightarrow \neg(E \vee L)$
c. $L \rightarrow \neg(M \vee E)$
d. $E \vee L \vee M$

[3] Though using his knowledge that bare numerals are often used with an 'at least' interpretation when used distributively, this may be the preferred assumption.

[4] This follows from the example in (12) as well as the examples from Musolino's first two experiments.

(18) $\{\neg M \neg EL, M \neg E \neg L, \neg ME \neg L\}$

The scenario presenting three horses jumping over a fence is an observation made by the child that triggers an update of its current information state. This observation corresponds to E in (16). Together with the additional premise (17a), the new, updated, information state of the child is (19a), which is expressed by the sentence (19b). By contrast, this observation falsifies (19c) because it is not in accordance with the observation made by the child.

(19) a. $\{\neg ME \neg L\}$
b. Exactly three horses jumped over the fence.
c. Two of the horses jumped over the fence.

In the frame theory of [Pet07], one way of representing an event of type 'Three horses jumping over the fence' is given in Figure 1.

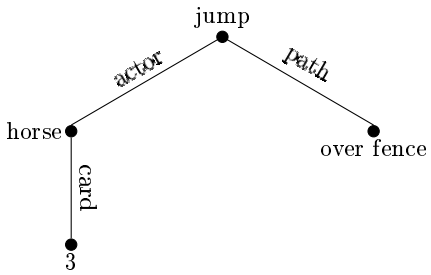

Fig. 1. Barsalou-Löbner frame

However, such a frame or feature structure only models *one* possible option of how the current information state of an agent can be updated upon hearing the utterance 'Three horses jumped over a fence'. It does not capture the agent's epistemic uncertainty about the fact that 'three' can have three different readings.

In our approach, this problem is solved by using a different form of representation that resembles the tabular form in database theory. Consider Table 3 ($\oplus$ is the operation which maps a sum-object in a Link-style representation to its set of atoms).

This table can be taken as a set of events with each event being a finite mapping s from a domain $\mathrm{dom}(s)$ to the universe U of a model M. Elements of $\mathrm{dom}(s)$ are called features or attributes. They can be properties of objects like colour or profession, but they can also represent properties of events like their type and thematic relations like actor. One way of interpreting this table is the following. It is split into three (sub-)tables such that each subtable matches one of the three possible readings allowed by 'three'.

	type	path	actor
e_1	jumping	f	$h_1 \oplus h_2 \oplus h_3$
e_2	jumping	f	$h_1 \oplus h_2 \oplus h_3 \oplus h_4$
e_3	jumping	f	$h_1 \oplus h_2 \oplus h_3 \oplus h_4 \oplus h_5$
e_4	jumping	f	$h_1 \oplus h_2$
e_5	jumping	f	h_1

Table 3. A frame for the event of three horses jumping over a fence

The set of events in Table 4 (left) corresponds to the reading 'Exactly three horses jumped over the fence'. The subtable consisting of e_1, e_2 and e_3 (Table 4, middle) represents the reading that at least three horses jumped over the fence (assuming a total of five horses) whereas the table made up of e_1, e_4 and e_5 corresponds to 'At most three horses jumped over the fence', Table 4 (right).

'exactly 3':

	path	actor
e_1	f	$h_1 \oplus h_2 \oplus h_3$

'at least 3'

	path	actor
e_1	f	$h_1 \oplus h_2 \oplus h_3$
e_2	f	$h_1 \oplus h_2 \oplus h_3 \oplus h_4$
e_3	f	$h_1 \oplus h_2 \oplus h_3 \oplus h_4 \oplus h_5$

'at most 3'

	path	actor
e_1	f	$h_1 \oplus h_2 \oplus h_3$
e_4	f	$h_1 \oplus h_2$
e_5	f	h_1

Table 4. Teams for modified bare numerals

Each situation represented by one of the three tables is epistemically indistinguishable for the agent because he does not know which reading applies, given only the information 'three horses'.

4 Strategies, teams, sequential composition and dynamic updates

As shown by the data in section 2.2, one major problem that has to be solved in analyzing scalar modifiers is the fact that their meaning cannot be reduced to a set-theoretical condition. We will use strategies to solve this problem. Intuitively, a strategy is a plan to reach a goal. In the linguistic case, a goal is determined by the truth conditions of an expression, for example the set-theoretic condition imposed by a scalar modifier. In general there can be different ways to reach a goal. For example, a superlative scalar modifiers like 'at least n' is analyzed as a strategy that allows to choose between two branches. The two branches correspond to splitting the set-theoretic condition in a deterministic ('=n') and a non-deterministic component ('>n'). By contrast, for the comparative modifier 'more than n-1' one only gets the non-deterministic component and has thus no choice between different branches in the strategy. The motivation behind this splitting is based both on cognitive and complexity considerations (see section

5.2). In the case of bare numerals and scalar modifiers, hoices in a strategy express a condition on the cardinality attribute on the NP of which they are part. Thus, they correspond to the available readings of those expressions. For example, an 'exactly'-reading requires a particular value of the cardinality attribute whereas for an 'at least'-reading no particular cardinality is determined although the set of admissible values of the cardinality attribute has to satisfy a specific condition: it has to be a filter. Strategies are introduced in section 4.1.

In our framework, such constraints on the cardinality attribute are modeled using techniques from Dependence Logic ([Vää07]). A central notion in Dependence Logic is that of a *team*, i.e. a set of assignments. A team represents one possible way the world could be according to the beliefs of an agent. Since formulas are interpreted in Dependence Logic not as sets of assignments but as *sets of sets of assignments*, it is possible to impose dependence relations on a team which must hold globally for the whole domain of the model. For example, a limiting case of functional dependence, namely constancy, is expressed by the formula $=(x)$, which says that the value of the attribute x is constant in a team. This formula will be used for 'exactly'-readings. When taken together, the meaning of a bare numeral or a scalar modifier is a pair consisting of a strategy and a set of teams with each team corresponding to one of the choices (branches). Together with structures formalizing such pairs, Dependence Logic is introduced in sections 4.2 and 4.3.

Combining two strategies is defined as sequential composition: each possible choice of the strategy corresponding to the modifying expression (say 'at least') is extended with every choice of the strategy denoted by the modified expression (say 'two'). Combining the set of teams corresponding to the different choices in a (branching) strategy are modeled as an update operation based on the notion of a supplement of a team (sections 4.4 and 4.5). The interaction of sequential composing and updating team decorated trees is illustrates in section 4.6.

4.1 Strategies for modeling different readings

Our definition of a strategy closely follows [PS11]. [PS11] distinguish basic and complex strategies. Complex strategies are built from basic ones using regular operations from Propositional Dynamic Logic (PDL) like sequencing ';', choice '$\cup$' and iteration '*'. Basic strategies can be branched. Branching is used to model the possibility for an expression of having more than one interpretation, like bare numerals for instance. Complex strategies are used to interpret modifiers like 'at least', which apply to bare numerals denoting basic strategies. The most important reason for using strategies is the following. Recall that 'at least n' and 'more than n-1' define the same set-theoretical relation although both expressions differ in meaning, as shown by the data in section 2.2. Using strategies, this difference can be explained as follows. Each strategy defines a set of states which can be reached by following it. Two strategies can differ although they determine the same set of states.

Strategies are defined in terms of finite labeled trees:

Definition 1 (Finite labeled tree) *Let Σ be a (non-empty) finite set of labels. A* Σ-labeled finite tree *T is a tuple $\langle S, \{\Rightarrow_a\}_{a\in\Sigma}, s^0\rangle$ where (i) S is a (non-empty) finite set of nodes, (ii) $s^0 \in S$ is the root of T and (iii) for each $a \in \Sigma$, $\Rightarrow_a \subseteq S \times S$ is the edge relation satisfying the usual properties of being irreflexive, antisymmetric and having a unique predecessor, i.e. if $s_1 \Rightarrow_a s$ and $s_2 \Rightarrow_b s$ then $s_1 = s_2$ and $a = b$.*

For a given node $s \in S$, the set $A(s) = \{a \in \Sigma \mid \exists s' \in S : s \Rightarrow_a s'\}$ is the set of *actions available (or executable) at s*. A *leaf node* is an element $s \in S$ s.t. $A(s) = \emptyset$. The set of all leaf nodes in a tree is denoted by $frontier(T)$. The root of a tree T is denoted by $root(T)$.

Definition 2 (Strategy tree) *A finite tree $T = \langle S, \{\Rightarrow_a\}_{a\in\Sigma}, s^0\rangle$ over a label set Σ is a (basic)* strategy tree *if its branching labeling is functional: for each $s, s', s'' \in S$ and $a \in \Sigma$, if $s \Rightarrow_a s'$ and $s \Rightarrow_a s''$ then $s' = s''$ ([PS11, 417]).*

Basic strategies are pairs consisting of a strategy tree and a global team, where the global team is a set of teams and each team is assigned to one leaf of the tree and vice versa (see section 4.2). The idea behind this definition can be explained as follows. The root node s^0 of a basic strategy tree is taken as a kind of *epistemic input* that triggers a particular plan to which an agent is committed if he has agreed to follow this strategy. Epistemic inputs can be observations by the agent (e.g. seeing a particular situation) or utterances by others. The global team assigned to $frontier(T)$ of an event model constitutes the new information with which the agent has to update his current information state (cf. [PS11]).

So far, we introduced strategy trees to model the different readings of bare numerals and scalar modifiers. However, if strategies are to be used as defining operations to update information states, the information associated with the leaf nodes of the tree cannot simply be taken to be 'indivisible'. Rather, the information given at those nodes must be such that it is possible to impose the cardinality constraints expressed by a choice in a strategy tree. Thus, two problems have to be solved: (i) which structures can be used to impose *global*, as opposed to *local*, constraints? and (ii) how can strategies as labeled trees be combined with such structures? The answer to the first problem is: (underspecified) teams from Dependence Logic. The second problem is solved by making use of the notion of a *team decorated tree*.

4.2 Dependence logic

The basic semantic notion used in Dependence Logic is that of a team, i.e. a set of assignments which map attributes (or variables) to elements of the domain of a first-order model.

Definition 3 (Team) *Let M be a first-order model, and let $\boldsymbol{v} = \langle v_1, \ldots, v_n\rangle$ be a tuple of variables. A* team *X for M with domain $\boldsymbol{v}$ is a set of assignments with domain $\boldsymbol{v}$ over M.*

In contrast to first-order logic, formulas are interpreted as sets of sets of assignments and, therefore, as sets of teams. Functional dependence between a sequence of attributes $x_1, \ldots, x_n$ and an attribute y is denoted by $=(x_1, \ldots, x_n, y)$. In addition to this dependence atom, the following two operators are defined which are similar to dependence atoms in being true of a team as a whole. $\uparrow x_n$ requires the values of the attribute x_n to be a filter and $\downarrow x_n$ requires the values to be an ideal.

(20) $M \models_X =(x_1 \ldots x_n, y)$ iff for all assignments $s, s' \in X$ with $s(x_i) = s'(x_i)$ for $i = 1, \ldots, n$, one has $s(y) = s'(y)$
$M \models_X \uparrow x_n$ iff $\forall s \in X : s(x_n) = \alpha \rightarrow \forall \beta(\alpha \sqsubseteq \beta \rightarrow \exists s' \in X : s'(x_n) = \beta)$
$M \models_X \downarrow x_n$ iff $\forall s \in X : s(x_n) = \alpha \rightarrow \forall \beta(\beta \sqsubseteq \alpha \rightarrow \exists s' \in X : s'(x_n) = \beta)$
($\sqsubseteq$ is the part-of relation in a Link-style representation of plural objects)

For formulas of First-Order Logic which do not contain a dependence atom, satisfaction with respect to a team X reduces to the usual Tarskian semantics in the sense that a formula is satisfiable just in case it is satisfiable with respect to each assignment in the team. For example, one has (21).

(21) $M \models_X \phi \wedge \psi$ iff $M \models_X \phi$ and $M \models_X \psi$.

Disjunction ($\otimes$) is defined on the basis of a split team:

(22) $M \models_X \phi \otimes \psi$ iff there exist Y and Z with $X = Y \cup Z$ such that $M \models_Y \phi$ and $M \models_Z \psi$.

Furthermore, we need two operators comparing the values of attributes:

(23) $M \models_X (x_1 = x_2)$ iff $\forall s \in X : s(x_1) = s(x_2)$
$M \models_X (x_1 < x_2)$ iff $\forall s \in X : s(x_1) \sqsubseteq s(x_2)$

So far we have shown that in Dependence Logic it is possible to impose global constraints on a team such as they are expressed by the various branches of a strategy. What is missing is a combination between strategies as labeled trees and teams. This link is defined in terms of team decorated trees which are a variant of feature decorated trees ([BGWV93]).

Definition 4 (Team decorated finite labeled tree; [BGWV93, 24]) *A* finite team decorated tree *is a pair $\langle T, D\rangle$ where T is a finite labeled tree and D is a function that assigns to each element of $frontier(T)$ a team.*

Definition 5 (Basic strategy) *A* basic strategy *is a finite team decorated tree $\langle T, D\rangle$ where T is a strategy tree.*

Thus, each reading of an expression, represented by a choice in the corresponding strategy, is related to a team. If $frontier(T)$ consists of n elements, one gets a total of n teams. Each team in this set represents one possible way the world (context, situation) could be according to the beliefs (knowledge) of an agent, i.e. it is a (partial) description of how the world (context, situation) could possibly

be according to the agent.[5] When taken together, the union of these teams represents the agent's epistemic or doxastic uncertainty. Such sets modeling the information state of an agent are called *global teams* and are denoted by $\mathcal{X}$.

Definition 6 (Global team) *A* global team *based on a first-order model M is a set $\mathcal{X} = \{X_1, X_2, \ldots\}$ of teams based on M over the same signature.*

In the next section the team decorated trees for bare numerals, scalar modifiers, common nouns and observations will be defined. In addition, it will be shown how the relation between the labels on branches of strategies and teams can be formally defined.

4.3 Basic strategies for bare numerals,scalar modifiers, common nouns and observations

We start by defining the basic strategy for bare numerals (see also Figure 2).

Definition 7 (Basic strategy for a bare numeral) *A strategy for a bare numeral is a team decorated finite labeled tree of height 2 which is based on the label set $\Sigma = \{\pi^n, \pi^=, \pi^\geq, \pi^\leq\}$ with path set $Path = \{\pi^n\pi^=, \pi^n\pi^\geq, \pi^n\pi^\leq\}$. The root node of the tree is the expression whose meaning is determined by the strategy.*

The path prefix π^n of all paths expresses that an agent first fixes the base cardinality n. Each $a \in \{\pi^=, \pi^\geq, \pi^\leq\}$ corresponds to an option (or choice) an agent has when processing or interpreting the expression: $\pi^=$ is the operation that adds the constraint $=(card)$ (leading to an 'exactly'-reading), $\pi^\geq$ adds $\uparrow card$ ('at least'-reading) and $\pi^\leq$ adds $\downarrow card$ ('at most'-reading). The satisfaction clauses for the strategy labels are given in (24).

(24)

$$\pi^n:\ M \models_X base = \bigoplus_{i=1}^{n} type \qquad \pi^=:\ M \models_X =(card)$$

$$\pi^\geq:\ M \models_X \uparrow card \qquad \pi^\leq:\ M \models_X \downarrow card$$

$$\pi^>:\ M \models_X \uparrow card \wedge (base < card) \qquad \pi^<:\ M \models_X \downarrow card \wedge (card < base)$$

The strategy for a bare numeral is depicted in Figure 2. The teams at the leaf nodes are underspecified: First, the value can be the most general one. This is the case for the *type* attribute which is assigned the top element $\top$. Second, the value of the *base* attribute fixes the base cardinality ($\pi^n : M \models_X base = \bigoplus_{i=1}^{n} type$), that is an underspecified sum object of length n consisting of n 'things' dependent on the value of *type* (a similar argument applies to the *card* attribute, which is a complex attribute whose value is computed by the value of the *base* attribute).[6]

[5] Note that single teams can express uncertainty too. This is the case whenever the values of an attribute form a filter or an ideal. However, this uncertainty is due to the interpretation of the expression and need not arise from epistemic uncertainty.

[6] Here, we implicitly assumed the initialization assumption which will be introduced in (25) in section 4.4.

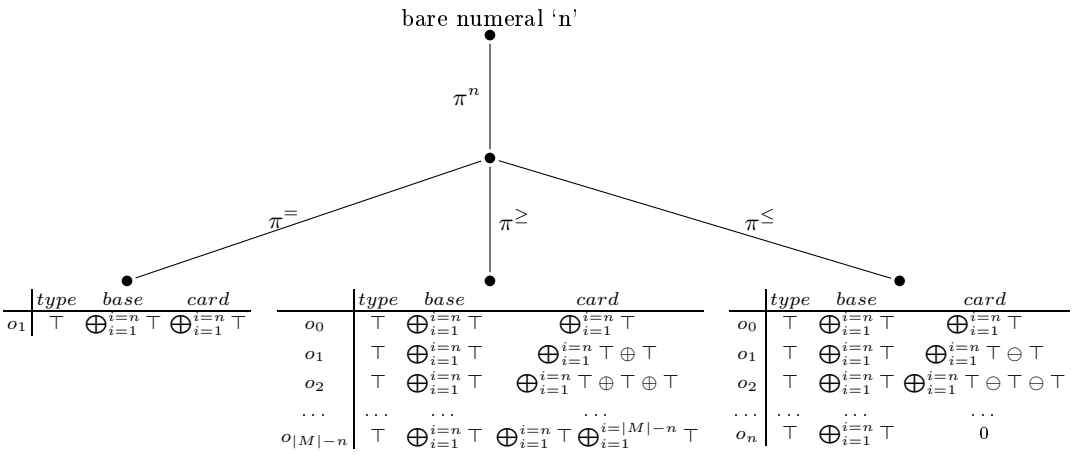

Fig. 2. Strategy for a bare numeral

The strategies for 'at least' and 'more than' are given in Figure 3. The basic strategy interpreting the superlative scalar modifier 'at least' is branching, i.e. it allows two different choices (or options). Either the cardinality information in the team is constant and therefore satisfies the constancy dependence atom $=(card)$, or the value of this attribute can vary and forms a filter, $\uparrow card$. By contrast, for 'more than', there is only the filter condition but no constancy requirement. If a bare numeral is combined with such a modifier, this is interpreted as a non-deterministic supplement operation (similar to the existential and the universal quantifier). Each choice that is possible for the modifier is combined with each choice that is admissible for the bare numeral (see section 4.4 below for details).

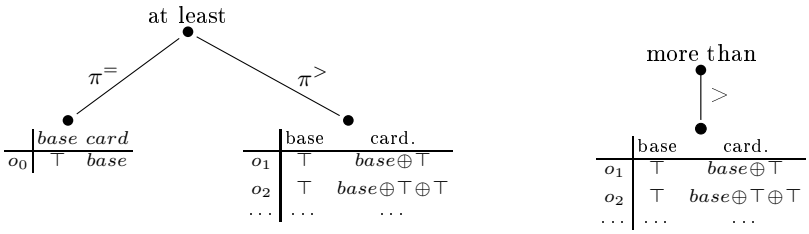

Fig. 3. Strategies for 'at least'(left) and 'more than' (right)

The meaning of common nouns like 'horse' are non-branching strategies of height 1. The label set Σ is a singleton and the only label *type* corresponds to the operation which fixes the value of the *type* attribute. The tree is given in Figure 4.

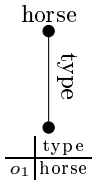

Fig. 4. Strategy for the common noun 'horse'

In contrast to the strategy for a bare numeral or another linguistic expression, the strategy for an observation is a tree of height 0. Thus, it has no labeled branches at all. Rather, the only node of the tree is assigned a formula (or a 1-element team) which expresses the content of the observation. In contrast to a strategy representing the meaning of a linguistic expression, a strategy corresponding to an observation admits of only one corresponding team which, furthermore, consists of only one assignment since its meaning is not underspecified.[7]

Definition 8 (Strategy for an observation) *A strategy for an observation P is a finite tree of height 0 with $\Sigma = \emptyset$. The associated team onsists of one assignment which specifies the content of the observation.*

In the present context, P is a formula of Dependence Logic which expresses a *global* property of a team, i.e. its truth cannot be reduced to its truth at single elements of the team.

Taking stock, we have shown how the meanings of bare numerals, scalar modifiers and common nouns can be modeled as team decorated trees consisting of a strategy the choices (branches) of which represent the different readings and the global team constituting the 'decorations' of the leaves modeling the object as a team satisfying the cardinality constraint imposed by the corresponding branch. What is missing so far is a mechanism of how such structures can be combined. Since those structures consist of a strategy and a global team, both components must be combined. Composing strategies is defined as a form of sequencing (section 4.4) whereas the combination of the global teams is defined as an update operation that is based on the notion of a supplement of a team (section 4.5).

4.4 Sequencing of two strategies

A scalar modifier combines with a bare numeral to form a more complex expression. In our framework, this operation is modeled by sequential composition.

[7] This does *not* mean that the observation is specific in the sense that an attribute is assigned a unique value. For example, without counting the number of horses that one sees one can say that one is seeing at most / at least n horses.

Definition 9 (Sequential composition of finite labeled trees) *Let* $T_1 = \langle S_1, \{\Rightarrow_a^1\}_{a\in\Sigma_1}, s_1^0\rangle$ *and* $T_2 = \langle S_2, \{\Rightarrow_a^2\}_{a\in\Sigma_2}, s_2^0\rangle$ *be two finite labeled trees with* $S_1 \cap S_2 = \emptyset$. *The sequential composition of* T_1 *and* T_2, *denoted by* $T_1;T_2$, *is the tree* T *in which each leaf of* T_1 *is replaced by a copy of* T_2.
Let $frontier(T_1) = \{f_1, f_2, \ldots, f_n\}$. *Then* $T = \langle S, \{\Rightarrow_a\}_{a\in\Sigma}, s^0\rangle$ *where*

(i) $S = S_1' \cup S_2'$ *with* $S_1' = \{(s,0)|s \in S_1\}$ *and* $S_2' = \bigcup_{1\le i\le n}\{(s,i)|s \in S_2 \setminus \{s_2^0\}\}$ *that is* S_2' *is the n-fold disjoint union of* $S_2 \setminus \{s_2^0\}$ *with* $n = |frontier(T_1)|$,
(ii) $s^0 = (s_1^0, 0)$,
(iii) $(s,i) \Rightarrow_a (s',j)$ *iff*
  a) $i = j$ *and* $s \Rightarrow_a^1 s'$ *or* $s \Rightarrow_a^2 s'$ *or*
  b) $i = 0$ *and* $j \neq 0$ *and* $s = f_j$ *and* $s_2^0 \Rightarrow_a^2 s'$.

According to Definition 9, the sequential composition of two trees T_1 and T_2 is construed by pasting (a copy of) the tree T_2 at all leaf nodes of T_1.

Let us first consider combining 'at least' with 'two'. In this case copies of the tree of the strategy representing 'two' are glued at the leaves of the strategy for 'at least'. The result of this sequential composition yields the tree in Figure 5(bottom left) of height 3 with six leaves, that is with a total of six theoretical choices.

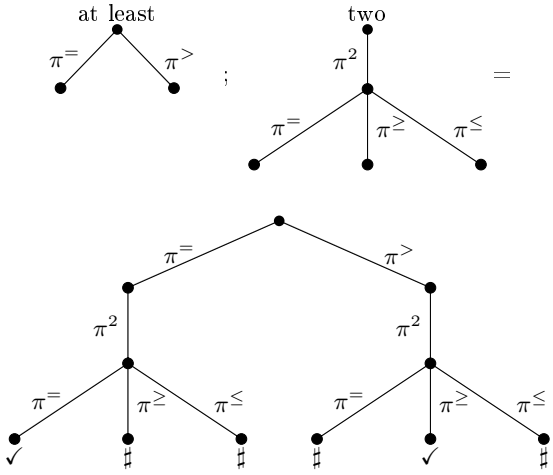

Fig. 5. Sequential composition of 'at least' and 'two'

In general, not all combinations of strategy labels along a path in a sequential composition tree yield satisfiable satisfaction clauses. Thus, in the case of a modified bare numeral not all strategies of the numeral may be admissible because they have to satisfy the constraint on the cardinality attribute imposed by the modifier. The following constraint relates the *cardinality* attribute to the *base* attribute if it is not initialized otherwise:

(25) **Initialization assumption:** If the *card* attribute is not initialized (as in the case of scalar modifiers) then

$$(card = base) \otimes =()$$

holds, i.e. there is at least one assignment in the team such that $card = base$ is satisfied. This condition is called the *initialization assumption*.

The satisfaction clauses along a path in a tree resulting from a sequential composition of two strategy trees is the conjunction of the clauses for the individual strategy labels in (24) plus the initialization assumption if it is necessary. Thus, the admissibility conditions for 'at least two' are calculated as follows (the initialization assumption is given in square brackets and contradictions are marked by #).

(26)
a. (✓) $\pi^{=}\pi^{2}\pi^{=} : M \models_X =(card) \wedge (base = \top \oplus \top)$
$[\wedge(card = base) \otimes =()]$
b. (#) $\pi^{=}\pi^{2}\pi^{\geq} : M \models_X =(card) \wedge \uparrow card \ \wedge \ldots$
c. (#) $\pi^{=}\pi^{2}\pi^{\leq} : M \models_X =(card) \wedge \downarrow card \ \wedge \ldots$
d. (#) $\pi^{>}\pi^{2}\pi^{=} : M \models_X \uparrow card \wedge (card > base) \wedge \ =(card) \ \wedge \ldots$
e. (✓) $\pi^{>}\pi^{2}\pi^{\geq} : M \models_X \uparrow card \wedge (base = \top \oplus \top) \wedge (card > base)$
f. (#) $\pi^{>}\pi^{2}\pi^{\leq} : M \models_X \uparrow card \wedge (card > base) \wedge \downarrow card \ \wedge \ldots$

Consider first 'at least'. The left choice imposes a constancy condition. This condition is satisfied on an 'exactly'-reading of the bare numeral (leftmost hoice, $\pi^{=}$) but not by the other two choices which impose either a filter ($\pi^{\geq}$) or an ideal ($\pi^{\leq}$) condition both of which violate constancy. If instead the other choice of the strategy for 'at least' is chosen ($\pi^{>}$), this filter condition excludes both the constancy and the ideal condition of the bare numeral. Non-satisfiable conditions yield strategies that can be removed from a strategy tree. Thus, in the example of 'at least two', the two remaining strategies together yield the strategy represented in Figure 6.

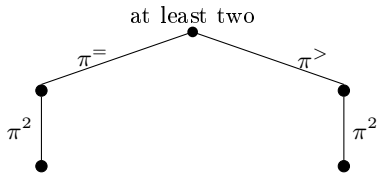

Fig. 6. Strategy for 'at least two'

For 'more than two', the argument goes as follows. The modifier 'more than' requires a filter condition for its cardinality attribute. Sequential composition with the bare numeral 'two' yields a tree with three leafs and thus three options which are shown in Figure 7 (below left). Two of these options are not admissible

because the filter condition imposed by the modifier is violated. Figure 7 (below right) shows the simplified resulting strategy for 'more than two'.

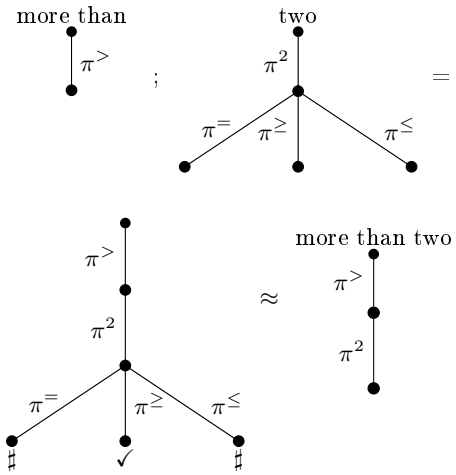

Fig. 7. Sequential composition of 'more than' and 'two'

Given these constraints, it follows that an agent only needs to initialize the attribute *base* if a scalar modifier is combined with a bare numeral.

4.5 Dynamic updates

The processing of utterances changes the input state of an agent. Let $\mathcal{X}$ be such an information state, i.e. a global team. Then an utterance α transforms $\mathcal{X}$ into $\mathcal{X}'$. If the update corresponding to α is $[\alpha]$, one gets $[\alpha]\mathcal{X} = \mathcal{X}'$. Since utterances are syntactically built up from their constituents, the update α must be defined in terms of more basic updates according to a finite set of operations. For example, 'at least n' is built from 'at least' and 'n'.

In this article, only one type of dynamic update is considered: expansive update. It applies to the set of teams making up the leaves of a team decorated tree. The basic idea is that each element of this global team must be updated by each element assigned to a leaf node (element of $frontier(T)$) of the strategy denoting a linguistic expression or an observation.

Formally, this is defined in terms of the notion of *supplementation of teams* ([Vää07]). The supplement operation on teams adds a new feature to the agents (or events) in a team, or alternatively, it changes the value of an existing attribute. The supplement operation on teams is formally defined as follows.

Definition 10 (Strict supplement of a team) *If M is a set, X is a team with M as its codomain and $F : X \to M$, $X(F/x_n)$ denotes the* supplement team $\{s(F(s)/x_n) : s \in X\}$.

$s(F(s)/x_n)$ is based on the notion of a modified assignment. If s is an assignment, then $s(a/x)$ is the assignment which agrees with s everywhere except that it maps x to a: $\mathrm{dom}(s(a/x)) = \mathrm{dom}(s) \cup \{x\}, s(a/x)(x) = a$ and $s(a/x)(x') = s(x')$ if $x' \neq x$ for $x' \in \mathrm{dom}(s)$. In a strict supplementation, the current team is expanded by assigning to each agent or event a single value for the attribute x_n.

An alternative way of defining the supplement operation consists in allowing that an agent or event can be assigned different values for the attribute x_n. This is the case whenever an element of $frontier(T)$ is not a singleton.[8]

Definition 11 (Lax supplement of a team) *If M is a set, X is a team with M as its codomain and $F : X \to \wp(M) \setminus \emptyset$, $X[x_n \mapsto F]$ denotes the* supplement team *of all assignments $s(a/x_n)$ with $a \in F(s)$.*

The following definition extends the previous one by allowing the supplementation of more than one attribute. The supplement of a team X' by a team X results in a team the domain of which is the union of the domains of X' and X. The supplement team keeps all information of X' and extends it with information about attributes which belong to the domain of team X but not of team X'. The latter attributes are successively supplemented to team X':

Definition 12 (Supplement of a team by a team) *Let X, X' be two teams. For $x_n \in dom(X)$ let F_{X,x_n} be the constant function that maps each $s \in X$ to $\{s(x_n) : s \in X\}$. Furthermore let $dom(X)$ be the domain of X and $infdom(X) = \{x \in dom(X) \mid \exists s \in X : s(x) \neq \top\}$ be the set of informative attributes of X. If $dom(X) \setminus infdom(X') = \{x_1, \ldots, x_n\}$, the supplement of a team X' by a team X is denoted as $\Delta_X(X')$ and defined as follows: $\Delta_X(X') = (\ldots((X'[x_1 \mapsto F_{X,x_1}])[x_2 \mapsto F_{X,x_2}])\ldots)[x_n \mapsto F_{X,x_n}]$*

Note that the former definition could be also read as an update of team X by team X': All attributes which are only defined in X are kept and extended by attributes which are unique for X'. For attributes which occur in the domain of both teams the information about admissible values given in the updated team X is overwritten by the updating team X' if the attribute is informative in X' (that is not constantly of the unspecific value $\top$).

A small example will illustrate the supplement operation and demonstrate that it is an operation which leads to an immense information loss. Consider the following two teams:

$X =$

	type	color	form
s_1	apple	red	round
s_2	peach	orange	round

and $X' =$

	type	color	taste
s_1	$\top$	red	sweet
s_2	$\top$	green	sour

The former could result from the observation of some round fruit that is either a red apple or an orange peach the latter from the thumb rule that the color of something is an indicator of its taste. If team X' is supplemented by team X (or alternatively X is updated by X') one gets:

[8] For details on the distinction between strict and lax semantics in Dependence Logic, see [Gal12].

$\Delta_X(X') =$

	type	color	taste	form
s_1	apple	red	sweet	round
s_2	peach	red	sweet	round
s_3	apple	green	sour	round
s_4	peach	green	sour	round

We have lost the information about the dependency between the attributes *color* and *type* in team X and the admissible value *orange* for the attribute *color*. Thus, as a general update operations on teams in a team-based semantics, definition 12 needs a careful revision. However, for our purpose here of combining strategies this rather coarse-grained update operation is sufficient if we filter the resulting teams by the satisfaction clauses resulting from the sequential combination of the strategies.[9] This will be the topic of the following section.

4.6 Putting update and sequencing together

We start by explaining how sequential composition works in parallel with an expansive update of the teams decorating the leaves. Suppose we have two strategy trees T_1 and T_2 which are sequentially combined $(T_1;T_2)$. Since each leaf node k of T_1 is replaced by a copy of the tree T_2, k is in effect replaced by $n = card(frontier(T_2))$ new leaves. Let $\Psi(k)$ denote the set of all teams decorating one of the n leaves which replace k. Let $X_{k,j}$ be the element of $\Psi(k)$ decorating the jth leaf of the tree T_2 and X_k be the team assigned to k in T_1. Furthermore let $C_{k,j}$ be the combined satisfaction clause resulting from the strategy labels along the path from the root node of the sequentially combined tree $T_1;T_2$ to the jth leaf of the copy of T_2 replacing k. Recall from section 4.4 that the strategy labels given in (24) are combined by conjunction plus the initialization assumption if necessary (see Figure 6 for an example). The team $X_{k,j}$ is transformed to a team X by supplementing it by team X_k (or alternatively, X_k is updated by $X_{k,j}$). The resulting team is only kept if it passes the filter imposed on it by the satisfaction condition $C_{k,j}$.

Let us first illustrate this construction by an example in which a bare numeral is combined with a common noun. An agent first fixes the base cardinality $(\pi^n : M \models_X base = \bigoplus_{i=1}^{n} type\,)$, that is an underspecified sum object of length n consisting of n underspecified 'things' described by *type* (say π^2 in the case of 'two'). There are three choices: 'exactly', 'at least' and 'more than' with the corresponding teams satisfying the condition imposed on the cardinality attribute. Next sequential composition is applied. Since the strategy for a common noun is non-branching (see Figure 4), each leaf node k of T_1, the strategy tree for the bare numeral, is replaced by exactly one leaf so that $\Psi(k)$ is a singleton for $1 \leq k \leq 3$. One therefore has three teams X_1, X_2 and X_3, resulting from

[9] A more fine-grained update or supplement operation can be defined if one uses a sort hierarchy on the values of an attribute. Using such a hierarchy, the value of the *type* attribute in the above example would be calculated as the greatest lower bound of the values in the corresponding elements of the teams.

T_1 and three teams $X_{1,1}$, $X_{2,1}$ and $X_{3,1}$ resulting from T_2. The team $X_{k,1}$ is transformed to a team X by supplementing it by team X_k. In this case there is no loss of information because in X_k the value of this attribute is $\top$, i.e. the most general information which is subsumed by the information provided by the common noun. Since $X_{k,1}$ does not impose any further satisfaction constraint, no team resulting from the supplement operation is discarded. For 'two horses', the resulting team decorated tree is given in Figure 8.[10]

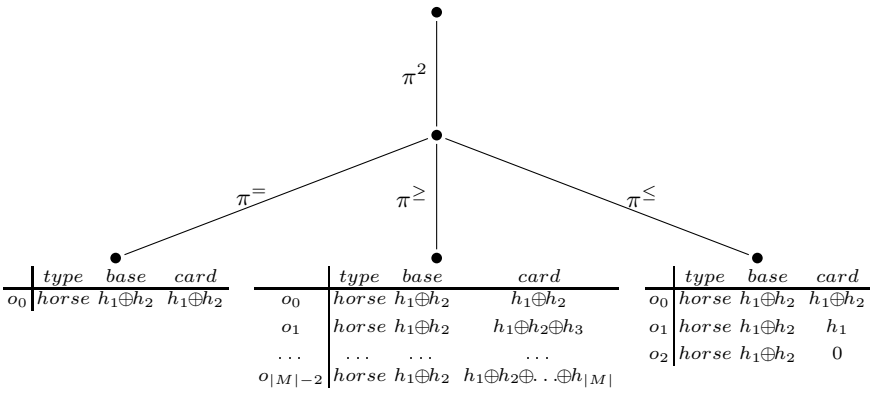

Fig. 8. Combined strategy for 'two horses'

Table 5 shows the information state of an agent after processing the expression 'two horses' and choosing the 'at least'-reading strategy.

	type	*base*	*card*
o_0	*horse*	$h_1 \oplus h_2$	$h_1 \oplus h_2$
o_1	*horse*	$h_1 \oplus h_2$	$h_1 \oplus h_2 \oplus h_3$
...	...	...	...
$o_{\|M\|-2}$	*horse*	$h_1 \oplus h_2$	$h_1 \oplus h_2 \oplus \ldots \oplus h_{\|M\|}$

Table 5. Information state after processing 'two horses' by hoosing the 'at least'-reading

Consider next the combination of 'at least' with 'two'. In this case $k = 2$ for the strategy tree corresponding to 'at least' (see Figure 3(left)). Applying sequential composition, yields a total of six branches since the strategy tree for 'two' has

[10] The indices at *horse* are used for better readability. Actually, *horse* has to be used without indices because the value of the *base* attribute is a Link-sum object over the values of the *type* attribute.

three leaves. Consequently, $\Psi(k)$ consists of three teams $X_{k,j}$ with $1 \leq k \leq 2$ and $1 \leq j \leq 3$. Since both strategy trees impose a satisfaction condition, $C_{k,j}$ is the combined satisfaction clause resulting from the strategy labels along the path from the root node of the sequentially combined tree $T_1;T_2$ to the jth leaf of the copy of T_2 replacing k. As was already shown in section 4.4, from the six paths of $T_1;T_2$ four are pruned because the corresponding satisfaction clause cannot be satisfied. Next the $X_{k,j}$ are transformed to a team X by supplementing them with team X_k. Since the attributes *base* and *cardinality* are common to $X_{k,j}$ and X_k, the values of these attributes in $X_{k,j}$ are kept. This does not result in a loss of information because, first, the value of the *base* attribute of the bare numeral is always more specific than the value of this attribute in X_k, where it is $\top$. Second, for 'at least', the value of the *cardinality* attribute is determined in terms of the value of the (completely underspecified) value of the *base* attribute. E.g., for the path $\pi^{>}$ its value for o_1 is $\mathit{base} \oplus \top$. For 'two', the corresponding value is $\bigoplus_{i=1}^{i=2} \oplus \top$. Since $\bigoplus_{i=1}^{i=2} \top$ is more specific than $\mathit{base} = \top$, there is again no loss of information.

The teams at the leaves of the remaining two non-pruned paths are computed as follows. The first one is the result of supplementing the team at the left leaf in Figure 2 ($\pi^2\pi^{=}$ strategy for 'two') with the team at the left leaf in Figure 3 ($\pi^{=}$ strategy for 'at least'):

	type	*base*	*card*
o_1	$\top$	$\top \oplus \top$	$\top \oplus \top$

The second one is the result of supplementing the team at the middle leaf in Figure 2 ($\pi^2\pi^{\geq}$ strategy for 'two') with the team at the right leaf in Figure 3 ($\pi^{>}$ strategy for 'at least'):

	type	*base*	*card*
o_1	$\top$	$\top \oplus \top$	$\top \oplus \top \oplus \top$
o_2	$\top$	$\top \oplus \top$	$\top \oplus \top \oplus \top \oplus \top$
...	...	...	...

If the expression 'at least two' is combined with a common noun like 'horse', the *type* attribute and consequently the dependent *base* and *ard* attributes will be further specified as described above.

5 Explaining the empirical data from section 2.2

5.1 Referential anaphora

Recall from section 2.2 the following difference between superlative and comparative scalar modifiers.

(27) a. I will invite at least two people, namely Jack and Jill.
b. ?I will invite more than one person, namely Jack and Jill.

Whereas superlative scalar modifiers allow a reference to specific objects using the 'namely'-construction, this is not the case for the corresponding comparative

scalar modifier. This difference is explained in terms of differences in the strategies for superlative and comparative modifiers. Recall that the strategy for the superlative scalar modifier 'at least' allows two different choices. Either the cardinality information in the team is constant and therefore satisfies the constancy dependence atom $=(card)$, or the value of this attribute can vary and forms a filter, $\uparrow card$. By contrast, for 'more than', there is only the filter condition but no constancy requirement. Thus, in contrast to the strategy for a comparative scalar modifier, the strategy for a superlative scalar modifier contains a *deterministic* element: $\pi^{=}$. Deterministic means that for any given state in a strategy exactly one transition applies (is possible). In PDL, the use of iteration (*) and choice ($\cup$) leads to non-determinism, if no restrictions are imposed. Restricting their use to contexts which yield deterministic programs (while-loops and the 'if then else' construct) is called *Strict Deterministic PDL* (SDPDL) ([HR83]). SDPDL is less complex than PDL. Its decision problem is in polynomial space whereas that of PDL is complete in deterministic exponential time. This leads to the following thesis: The 'namely'-construction requires a deterministic substrategy. In addition, the above way of defining the meaning of bare numerals and scalar modifiers explains their difference in cognitive complexity as follows: (i) bare numerals are simplest because they do not involve the composition of two strategies; (ii) 'At least' is more complex than 'more than' because it involves two different operations in a particular order: sequencing plus choice as opposed to only choice.

5.2 Explaining the acquisition data: 5-year-olds vs. adults

Similar to adults, 5-year-old children who are preschoolers already have implicit semantic knowledge of the non-deterministic interpretation of bare numerals. That is, they know that the interpretation of these expressions depends on the context and they are able to match these different interpretations successfully in the sense that they are able to perform observation updates after processing a sentence with a bare numeral. By contrast, their application skills do not yet match the accuracy of those shown by adults.

We conjecture that there are the following differences in the semantic knowledge of those children and adults. First, children aged 5 only master simple updates that are given by strategies. Although they already know that a bare numeral allows different readings, they need a context in order to decide which reading applies. They do *not* yet master complex strategies like those imposed by comparative and superlative scalar modifiers. The experiment of [GKC^{+}10], see section 2 above, shows that by the age of eleven, children know that comparative and superlative scalar modifiers impose strategies on the interpretation provided by bare numerals. However, their semantic knowledge shows a clear difference between monotone increasing scalar modifiers like 'at least' and monotone decreasing ones like 'at most' and 'fewer than'. This difference can be explained as follows. Consider the examples in (28) taken from [Sza10, 56].

(28) a. *At least two men walk* = There is a set of men with cardinality at least two such that all its elements walk.
b. *At most two men walk* ≠ There is a set of men with cardinality at most two such that all its elements walk.

If in a given situation two men walk, (28a) is true even if there is a larger situation in which further men walk. The same does not hold for (28b). If someone sees Bill and John walking, (28b) is false if there are other men who are walking too. One way of solving this problem consists in imposing a *maximality* condition on monotone decreasing expressions. In Dependence Logic this difference between monotone increasing and monotone decreasing quantifiers is captured by explicitly introducing a maximality condition (see [Eng12] for details).

(29) a. $M \models_X Qx\phi$ iff there exists $F : X \to \wp(M)$ such that $M \models_{X(F/x)} \phi$.

b. for each $F' \geq F$ s.t. $F' : X \to \wp(M)$, if $M \models_{X(F'/x)} \phi$, then for all $s \in X : F(s) \in Q$ where $F' \geq F$ iff for every $s \in X : F(s) \subseteq F'(s)$.

In (29a) Q is a generalized quantifier (of type $\langle 1 \rangle$) and $X[F/x]$ is the supplement operation. In order to also apply to monotone decreasing quantifiers, it is not sufficient to only require that there is a function F such that $M \models_{X[F/x]\phi}$. Rather, the maximality condition in (29b) must be added.

The poor performance on monotone decreasing modifiers shows that children at the age of eleven still do not master the maximality condition which is imposed by this type of modifier. Finally, the differences in processing load between this type of modifier and the upward entailing ones shows that even for adults the semantic interpretation of the former is more costly than that of the latter. The differences are summarized in Table 6.

	5-year-old	11-year-old	adult
basic strategy for bare numerals	yes	yes	yes
composition operation for scalar modifiers	no	yes	yes
maximality condition for monotone decreasing expressions	no	no	yes

Table 6. Comparison of semantic knowledge

Generalizing the above discussion, one arrives at the following tentative hypotheses: (i) sequential composition is costly if it involves pruning, i.e. if admissibility conditions have to be applied. Note that this additional mechanism need not be applied if a bare numeral is combined with a common noun; (ii) the maximality condition, too, can be said to involve an admissibility condition: only those supplement operations are admissible which are maximal among all supplement operations. (i) and (ii) together yield thesis (iii): whenever a compo-

sition or an update operation involves an admissibility condition which restricts the operation, a higher processing load is triggered.

6 Summary

In this article we developed a formal theory of the Löbner-Barsalou frame hypothesis. The meaning of an expression is a team decorated tree, i.e. a pair consisting of a strategy tree and a set of teams. Each team represents a possible reading of an expression and therefore models one epistemic alternative of an agent. Combining frames is defined in terms of two operations, one for each of the two components of a frame. This framework therefore accounts for two interrelated issues in a formal theory of frames: (i) How can frames be formally modeled? and (ii) How can updates of frames be explicitly modeled?

References

[BGWV93] P. Blackburn, C. Gardent, and W.Meyer-Viol. Talking about trees. In *EACL*, pages 21–29, 1993.

[Car98] R. Carston. Informativeness, relevance, and scalar implicature. In Robyn Carston and Seiji Uchida, editors, *Relevance Theory: Applications and Implications*, pages 179–236. John Benjamins Publishing Co., Amsterdam, 1998.

[Eng12] F. Engström. Generalized quantifiers in dependence logic. *Journal of Logic, Language and Information*, 21:299–324, 2012.

[Gal12] P. Galliani. Inclusion and exclusion dependencies in team semantics – on some logics of imperfect information. *Annals of Pure and Applied Logic*, 163(1):68–84, 2012.

[GKC+10] B. Geurts, N. Katsos, C. Cummins, J. Moons, and L. Noordman. Scalar quantifiers: Logic, acquisition, and processing. *Language and Cognitive Processes*, 25(1):130–148, 2010.

[GN07] B. Geurts and R. Nouwen. At least et al.: The semantics of scalar modifiers. *Language*, 83:533–559, 2007.

[Hor89] L.R. Horn. *A natural history of negation*. Chicago UP, Chicago, 1989.

[HR83] J. Y. Halpern and J. H. Reif. The propositional dynamic logic of deterministic, well-structured programs. *Theor. Comput. Sci.*, 27:127–165, 1983.

[Mus04] J. Musolino. The semantics and acquisition of number words: integrating linguistic and developmental perspectives. *Cognition*, 93(1):1 – 41, 2004.

[Pet07] W. Petersen. Representation of concepts as frames. In Jurgis Skilters, Fiorenza Toccafondi, and Gerhard Stemberger, editors, *Complex Cognition and Qualitative Science*, volume 2 of *The Baltic International Yearbook of Cognition, Logic and Communication*, pages 151–170. University of Latvia, 2007.

[PM03] A. Papafragou and J. Musolino. Scalar implicatures: experiments at the semantics-pragmatics interface. *Cognition*, 86(3):253 – 282, 2003.

[PS11] E. Pacuit and S. Simon. Reasoning wih protocols under imperfect information. *The Review of Symbolic Logic*, 4:412–444, 2011.

[Sza10] A. Szabolsci. *Quantification*. CUP, Cambridge, 2010.

[Vää07] J. Väänänen. *Dependence Logic*. CUP, Cambrige, 2007.
[vB11] J. van Benthem. *Logical Dynamics of Information and Interaction*. Cambridge University Press, 2011.