

Intensionality and Coercion*

Michiel van Lambalgen and Fritz Hamm

November 22, 2002

1 Aspectual coercion as an intensional phenomenon

Ever since Vendler's famous classification of verbs with respect to their time schemata, linguists have distinguished at least four aspectual classes or Aktionsarten. These classes comprise states, activities, accomplishments, achievements and points¹, which are exemplified in (1) to (5).

- (1) *States*: know, love, be beautiful.
- (2) *Activities*: run, push a cart, draw.
- (3) *Accomplishments*: cross the street, build a house, draw a letter.
- (4) *Achievements*: begin, reach, arrive.
- (5) *Points*: flash, spot, blink.

Several linguistic tests were developed to distinguish aspectual classes. For example, adverbial modification with *for two hours* is possible with activities (John ran for two hours) but not with accomplishments (*John built a house for two years) or achievements. Temporal modification with the adverbial *in two hours* shows the reverse pattern. Usually only activities and accomplishments can occur in the progressive. Expressions like **knowing the answer* are ungrammatical. But there is also a crucial difference in the behaviour of activities and accomplishments with respect to their progressivised forms. From *John was pushing a cart* we can infer that John pushed a cart. However, the inference from *John was crossing the street* to *John crossed the*

*This paper will appear in R. Kahle (ed.), *Intensionality*, ASL Lecture Notes in Logic, AK Peters.

¹This category was introduced by Moens and Steedman, see [24].

street is invalid. This phenomenon was dubbed the “imperfective paradox” by Dowty².

Note, however, that the aspectual properties introduced above cannot be *lexical* properties of the respective verbs alone. For instance, *John drank* is certainly an activity and *John drank beer* is an activity too. However, *John drank a glass of beer* is not an activity but an accomplishment. Therefore, the choice of the object NP (a mass term versus an indefinite NP) is a determining factor for aspectual classes. The next example from [24] shows that Aktionsart is not fully determined prior to the sentence level. *Chapman arrived* is an achievement, as can be seen from the ungrammaticality of **Chapman arrived all night*. But if we choose a bare plural subject instead of a proper name as subject NP, we get the grammatical sentence *Visitors arrived all night*.

Therefore the choice of the object and subject NPs can force verbs to be interpreted in different aspectual classes. This reinterpretation process was dubbed *coercion* in [15]. Unfortunately, this term does not refer to a coherent class of phenomena. Consider example (6):

(6) Pollini played the sonata for two days.

Pollini played the sonata is an accomplishment and accomplishments are generally bad with for-adverbials as pointed out above. Nevertheless sentence (6) can mean that Pollini played the respective sonata repeatedly within a timespan of two days. Thus the accomplishment *Pollini played the sonata* is coerced to an iterative reading. But this time we cannot identify a linguistic constituent of the sentence which is responsible for this reinterpretation. The expression *for two days* does not force an iterative reading in *John slept for two days* and neither does the accomplishment *Pollini played the sonata*.

Worse, even cases of metonymy are often considered as instances of coercion. Thus a sentence like *The pianists are on the top shelf* interpreted with respect to a scene within a CD-shop means that the CDs of pianists are on the top shelf. The NP *The pianists* is in this context reinterpreted as *CDs of pianists* on the basis of a threatening type conflict with the VP *be on the top shelf*.

In this paper we will certainly not do justice to all phenomena which are grouped together under the linguistic term *coercion*, but we will concentrate on those which pertain to Vendler’s classification outlined above. Moreover,

²Many tests which we cannot mention here were devised by linguists. An almost comprehensive list is in [7].

we will not comment on the vast literature on reinterpretation but refer the interested reader to a selection of more recent work³.

In several quarters of the semantic world, Vendler's classification of verbs into states, achievements, activities and accomplishments is often conceived of as pertaining to inherent properties of a verb (or, rather verb phrase). By contrast, various forms of cognitive grammar treat aspect as a way to *impose* temporal structure on events, or a way to conceptualise the temporal constitution of events. In the words of Croft [4, p. 70]:

The aspectual grammatical constructions determine in part the temporal structure of the event it describes via conceptualization.

The difference between these ways of treating aspect is subtle but important, as can be made clear by means of an example due to Croft (*op. cit.*). If the verb *love* is considered to be inherently stative, then

(7) *I am loving her

is ungrammatical, and cognitive linguists would agree; here the intended content has to be expressed by

(8) I love her.

But if (7) is provided with a context, it may become grammatical, as in

(9) I am loving her more and more every day, the more I get to know her.

This phenomenon is hard to explain if stativity is taken to be a property of the verb *love*. It seems more profitable to explain the contrast by saying that the events referred to in (8) and (9) are conceptualised differently, so that the event taken as an atom in (8) is considered to have stages and phases in (9). Thus, the phenomenon of *aspectual coercion* comes to the fore: the potential of grammatical constructions, such as the progressive, to 'move' a verb from one aspectual category to another.

Note that in general we cannot expect to arrive at the result of such a 'move' in a compositional way. To see this, consider again example (6). As pointed out, the linguistic material itself does not give any clue that an iterative reading is prominent here. We have to rely on extralinguistic knowledge such as the timespan it usually takes to play a sonata to derive

³See especially [15], [24], [18], [19], [6], [5], [9]

this meaning. A theory of coercion effects should therefore allow to combine very different aspects of meaning within a unified formal framework.

Traditional accounts of intensionality have been very much concerned with objects (Morning Star/Evening Star) and propositions (φ in ‘I believe that φ ’). The above considerations suggest that intensionality also plays a prominent role when talking about events. In fact, Frege’s view of intensionality as *Art des Gegebenseins* fits the phenomena outlined above very well.

In modern formal semantics, Frege’s notion of *Art des Gegebenseins* is reconstructed via possible worlds. Therefore, intensional phenomena in semantics are those where possible worlds play an essential role for the description of their meaning. A quick glance at natural language constructions analysed with possible worlds in modern semantics reveals that this again is an extremely heterogenous domain. Here are two examples which demonstrate this diversity quite clearly. The first are propositional attitudes like *John believes that Pythagoras was looking for the rational $\sqrt{2}$* . Here the set of worlds which support the sentence *Pythagoras was looking for the rational $\sqrt{2}$* and are compatible with John’s belief are used to model John’s belief state.

For a progressivised accomplishment like *Pollini was playing the sonata* which is in general analysed as an intensional construction, the role of worlds is completely different. In Dowty’s by now classical analysis, worlds are used to model *inertia*.

Furthermore, the standard tests for identifying intensional constructions emphasise the observed difference. According to the first, a construction is *extensional* if equivalent expressions can be substituted *salva veritate*. An expression is intensional if there are exceptions to this substitutivity principle. Clearly propositional attitudes are intensional according to this principle. The progressive, on the other hand, is not intensional according to this test. From *Bill was painting Dan (Kavanaugh)* we can intuitively infer *Bill was painting Julian (Barnes)*, given that Dan (Kavanaugh) and Julian (Barnes) are the same person.

The second criterion says that intensional constructions are those for which the rule of existential generalisation fails. Again, propositional attitudes are intensional according to this criterion⁴. On the basis of this principle, the progressive, too, is classified as intensional. From *Mary was drawing*

⁴Neither principle is watertight. This was observed for the first principle in [21]. Consider the sentence

Clark Kent went into the phone booth, and Superman came out.

a *circle* we cannot infer that there exists a circle that Mary was drawing.

Other intensional constructions in natural languages, such as modals, infinitives, verbs like *seek* versus at first sight extensional verbs like *fetch* clearly demonstrate the empirical heterogeneity of the intensional domain.

The conclusion is that according to the practice of possible world semantics, *coercion* would certainly be considered an intensional phenomenon. However, given the empirical diversity on the one hand and the rather one-dimensional methodological approach of possible world semantics on the other, this seems to be a rather weak reason. Therefore, the clearest conceptual motivation for treating *coercion* as an intensional phenomenon rests on Frege's description of his concept *Art des Gegebenseins*.

2 A brief introduction to the event calculus

We need a formal system in which all the conceptualisations of events as they occur in natural language can be defined. To have an idea of the kind of structure that is necessary, it is useful to consider one of the most complex classes of verbs, called *accomplishments* by Vendler, of which examples are 'draw a circle', 'write a letter', 'cross the street'. Events representing such verbs have an elaborate internal structure. On the one hand there is an activity taking place (draw, write, cross), on the other hand an 'object' is being 'constructed': the circle, the letter, or the path across the street. Furthermore, each such event is associated with a natural consequent state, where a (complete) circle has been drawn, a letter has been written, or a street has been crossed. We may also remark that the activities involved are volitional and so come with events marking the beginning or end of the activity. The literature contains several formalisms for reasoning about events, which have their roots in planning systems in artificial intelligence. It has been suggested several times that such formalisms might be useful for the semantics of natural language, although [11] seems to be the first paper where the actual computations are done. We borrow the basic format from [22], although the computational tools will be different.

The above considerations suggest that at least the following ontology is necessary for talking about events:

Traditionally this is not thought of as an intensional construction; nevertheless *Clark Kent* cannot be substituted for *Superman* *salva veritate* in this sentence. The counterexample to the second principle is from [3]. The sentence *The rational $\sqrt{2}$ does not exist* is considered extensional, but existential generalisation applied to it results in the nonsense *There exists something such that it does not exist*.

1. individuals
2. real numbers, both to represent time and to code ‘stages’ of partial objects
3. time-dependent properties, such as activities
4. changing partial objects
5. events, marking the beginning and end of time-dependent properties.

Both time-dependent properties and changing partial objects can be brought under the heading of a *fluent*. A fluent is a function which may contain variables for individuals and reals and which is interpreted in a model as a set of time points. Events will be taken in the sense of event types, from which event tokens are obtained by anchoring the event type to a time point (sometimes also to an interval).

Before going on to a discussion of the primitive predicates, we must fix our usage of the term ‘event’. The attentive reader will have noticed that we have used ‘event’ in two senses: as a generic term, as when we said that events can be conceptualised in different ways, and in a more specific sense, as marking the beginning and end of time-dependent properties. In the context of the formal theory, events in the second sense are taken to be unstructured objects (possibly parametrised). Henceforth, events in the first sense will be called *eventualities*, following Bach’s usage in [1].⁵

Given this ontology, the following choice of basic predicates seems natural. We want to be able to say that events happen, that fluents are initiated and terminated by events, or that a fluent was true at the beginning of time. If f is a variable over fluents, e a variable over events, and t a variable over time points, we may write the required predicates as

1. *Initially*(f)
2. *Happens*(e, t)
3. *Initiates*(e, f, t)
4. *Terminates*(e, f, t)

⁵This distinction is not meant to be absolute. It could very well be that an event in the second sense does have interesting internal structure. The distinction is meant to be relative to the choice of a level of granularity, and the formal theory applies to a specific level only. It is a different matter entirely to model a transition between levels.

These predicates are to be interpreted in such a way, that if $Happens(e, t) \wedge Initiates(e, f, t)$, then f will begin to hold after (but not at) t ; if $Happens(e, t) \wedge Terminates(e, f, t)$, then f will still hold at t . Strictly speaking, this convention makes sense only for events which are not extended in time. For the general case, one needs some axioms additional to those presented below; we will omit these for the sake of simplicity.

The possibility of having changing partial objects requires its own special predicates, namely

5. $Trajectory(f_1, t, f_2, d)$
6. $Releases(e, f, t)$

We have seen above that an activity, while it is going on, may change a partial object. The first predicate in the above list embodies this. Here, one should think of f_1 as an activity and of f_2 as a certain stage of a partial object. The predicate then expresses that if f_1 holds from t until $t + d$, then at $t + d$, f_2 holds. In applications, f_2 will have a real argument and will be of the form $f_2(g(t + d))$ for some continuous function g . The predicate $Releases$ is necessary to cancel the effect of those axioms of the event calculus which intend to express the so-called ‘principle of inertia’⁶. These axioms have the form: if there are no f -relevant events between t_1 and t_2 , then the truth value of f at t_1 is the same as that at t_2 . We introduce two special predicates for f -relevant events. The first predicate expresses the idea that there is a terminating or releasing event between t_1 and t_2 ; the second predicate states that there is an initiating or releasing event between t_1 and t_2 .

7. $Clipped(t_1, f, t_2)$
8. $Declipped(t_1, f, t_2)$

Lastly, we need the truth predicate

9. $HoldsAt(f, t)$

In the usual set-up of the event calculus, it is only *said* that $HoldsAt$ is a truth predicate; the defining axioms for the truth predicate are lacking. In planning applications of the event calculus, fluents are typically derived from first order formulas in a language disjoint from that of the event calculus itself. ‘Derive’ here refers to an operation which transforms formulas into

⁶Which says: ‘unless there is explicit information to the contrary, we may assume an action does not affect a fluent’.

terms, for example, Gödelisation, or what in AI is termed reification. It is of course easy to declare a two-valued truth predicate applying to such fluents only. However, once the fluents are codes of formulas which may also contain predicates from the event calculus, this becomes problematic, and in that case one needs to add a logic program for the truth predicate to the event calculus. The proper logic for such a truth predicate is Kleene's strong three-valued logic. We do not know of any planning application which needs this generality, but, as explained in [11], for a semantics of natural language this generality is essential. Gödelisation or reification is the formal counterpart of the syntactic operation of nominalisation, and when this procedure is iterated, as in

(10) My father objecting to my not going to church

the truth conditions for sentences involving this expression involve nested occurrences of *HoldsAt* as well. These observations notwithstanding, in the interest of brevity and legibility we shall not add a truth theory, and we shall assume some external interpreter available which relates a fluent to the formula (not containing predicates of the event calculus) from which it derives.

All in all, we then have

Definition 1 *An EC-structure is a many-sorted structure of the form⁷ $(R \uplus D \uplus E \uplus F; 0, 1, +, \cdot; a_1 \dots a_k, f_1 \dots f_n, e_1 \dots e_m, \text{Basic})$, where*

1. *R denotes a real closed field, D a set of individual objects, E a set of events, F a set of fluents.*
2. *$a_1 \dots a_k$ are elements of D and interpret individual constants.*
3. *$f_1 \dots f_n$ are the interpretations of fluent functions (with variables for reals and individuals).*
4. *$e_1 \dots e_m$ are the interpretations of event functions (with variables for reals and individuals).*
5. *Basic is an interpretation of the basic predicates introduced above.*

Definition 2 *The predicates Initially, Happens, Initiates, Terminates, Trajectory and Releases are called the primitive predicates of the event calculus. (We consider HoldsAt to belong to the truth theory.)*

⁷ \uplus denotes disjoint union.

The axioms of the event calculus given below are modified from [22], the difference being due to the fact that we prefer a constraint logic programming approach, whereas Shanahan uses circumscription. In the following, all variables are assumed to be universally quantified.

Axiom 1 $Initially(f) \wedge \neg Clipped(0, f, t) \rightarrow HoldsAt(f, t)$

Axiom 2 $Happens(e, t) \wedge Initiates(e, f, t) \wedge t < t' \wedge \neg Clipped(t, f, t') \rightarrow HoldsAt(f, t')$

Axiom 3 $Happens(e, t) \wedge Terminates(e, f, t) \wedge t < t' \wedge \neg Declipped(t, f, t') \rightarrow \neg HoldsAt(f, t')$

Axiom 4 $Happens(e, t) \wedge Initiates(e, f_1, t) \wedge t < t' \wedge t' = t + d \wedge Trajectory(f_1, t, f_2, d) \wedge \neg Clipped(t, f_1, t') \rightarrow HoldsAt(f_2, t')$

Axiom 5 $Happens(e, s) \wedge t < s < t' \wedge (Terminates(e, f, s) \vee Releases(e, f, s)) \rightarrow Clipped(t, f, t')$

Axiom 6 $Happens(e, s) \wedge t < s < t' \wedge (Initiates(e, f, s) \vee Releases(e, f, s)) \rightarrow Declipped(t, f, t')$

The set of axioms of the event calculus will be abbreviated by *EC*. In the absence of further axioms, one can construct a model for *EC* by interpreting fluents as finite unions of halfopen intervals (of the form $(a, b]$) and assuming that each event either initiates or terminates a fluent and that fluents are initiated or terminated by events only. Interestingly, such models are also available in the presence of further sentences, laying down preconditions on *Happens* and the other primitive predicates. Indeed, this property will play an important role in the proposed aspectual calculus.

2.1 Scenarios

While the above axioms state what is generally true about fluents and events, the structure of specific eventualities must be described by micro-theories. For example, in the case of ‘draw a circle’ the eventuality is composed (at least) of an activity (‘draw’) and a changing partial object (the circle in its various stages of completion); the micro-theory should specify how the activity ‘draw’ is related to the amount of circle constructed. This is done by means of three definitions.

Definition 3 A state $S(t)$ at time t is a complex body (cf. definition 6) involving only

1. literals of the form $(\neg)\text{HoldsAt}(f, t)$, for t fixed and possibly different f .
2. equalities between fluent terms, and between event terms.
3. formulas in the constraint language \mathcal{L} .

Definition 4 A scenario is a conjunction of statements of the form

1. $\text{Initially}(f)$, or
2. $\forall t(S(t) \rightarrow \text{Initiates}(e, f, t))$, or
3. $\forall t(S(t) \rightarrow \text{Terminates}(e, f, t))$, or
4. $\forall t(S(t) \rightarrow \text{Releases}(e, f, t))$, or
5. $\forall t(S(t) \rightarrow \text{Happens}(e, t))$,

where $S(t)$ is a state in the sense of definition 3. These formulas may contain additional constants for objects⁸, reals or time points and can be prefixed by universal quantifiers over time points, reals and objects.

Definition 5 A dynamics is a statement of the form $S(f_1, f_2, t, d) \rightarrow \text{Trajectory}(f_1, t, f_2, d)$, where $S(f_1, f_2, t, d)$ is a state in the sense of definition 3.

3 Minimal models

When it comes to making inferences, the event calculus as presented so far is very weak, and definitely not a model for human reasoning about events. Common sense reasoning seems to be governed by the principle of inertia: ‘normally, an action does not affect a property of the world’. This requires a non-classical notion of validity, in which one does not look at *all* models of a set of premises, but only at *minimal* models, that is models in which the interpretations of the primitive predicates of the event calculus are as small as is consistent with those premises. In such models there are no spurious happenings of events which may nonetheless affect classical validity. For example, we would like to derive from a scenario for ‘cross the street’ that,

⁸We do not allow functions on objects.

barring unforeseen circumstances, one reaches the other side of the street. It is impossible to derive such a conclusion classically, because there will be a model in which lightning *happens* and has the effect of *terminating* the crossing. There are several ways of formalising this intuitive idea. Shanahan chose circumscription, we opt for constraint logic programming with (a version of) negation as failure, mainly because we want to have the reals as the underlying structure of time.

3.1 Logic programming with real-closed fields

Constraint logic programming is concerned with the interplay of two languages. Let \mathcal{L} be the language $\{0, 1, +, \cdot, <\}$, \mathcal{T} the complete theory of $(\mathbb{R}, 0, 1, +, \cdot, <)$ in \mathcal{L} , i.e. the theory of real-closed fields. Let \mathcal{K} be another language, consisting of programmed predicate symbols. The constraint programming language $CLP(\mathcal{T})$ consists on the one hand of *constraints*, which are first order formulas from the language \mathcal{L} , and on the other hand of formulas from \mathcal{K} , whose terms come from \mathcal{L} .⁹ Constraint logic programs differ from logic programs by allowing constraints in the bodies of rules and in queries. For example, primitive constraints in $CLP(\mathcal{T})$ include formulas of the form $s < t$ and $s = t$, where s, t are terms from \mathcal{L} . *Definite* constraint logic programs have the form

$$B_1 \wedge \dots \wedge B_m \wedge c \rightarrow A,$$

where B_i and A are atoms in \mathcal{K} and c is a constraint. Likewise, a query has the logical form

$$B_1 \wedge \dots \wedge B_m \wedge c \rightarrow \perp.$$

We shall use the notation

$$?c, B_1 \dots B_m$$

for queries, always with the convention that c denotes the constraint, and that the remaining formulas come from \mathcal{K} . The words ‘query’ and ‘goal’ will be used interchangeably.

The aim of a constraint computation is to express a programmed predicate symbol entirely in terms of constraints. Thus, unlike the case of ordinary logic programming, the last node of a successful branch in a computation tree contains a constraint instead of the empty clause.

For our purposes, definite constraint logic programs are not yet expressive enough. We follow [25] in allowing (classical equivalents of) arbitrary

⁹We shall slightly relax this condition later.

first order formulas in the body of program clauses. More precisely, we define a *complex subgoal* recursively to be

1. an atom in \mathcal{K} , or
2. $\neg\exists\bar{x}(B_1 \wedge \dots \wedge B_m \wedge c)$, where c is a constraint and each B_i is a complex subgoal.

Definition 6 A complex body is a conjunction of complex subgoals. A normal program is a formula $\phi \rightarrow A$ of $CLP(\mathcal{T})$ such that ϕ is a complex body and A is an atom.

If, in the second clause of the above definition, we take \bar{x} to be empty, we obtain ordinary goals (with constraints), which as indicated will be written as $?c, B_1 \dots B_m$.

The interpretation of negation most congenial to constraint logic programming is *constructive* negation ([25]). In the customary negation as failure paradigm, negative queries differ from positive queries: the latter yield computed answer substitutions, the former only the answers ‘true’ or ‘false’. Constructive negation tries to make the situation more symmetrical by also providing computed answer substitutions for negative queries. Applied to constraint logic programming, this means that both positive and negative queries can start successful computations ending in constraints. As in the case of negation as failure, the proper logic for negation in constraint logic programming is Kleene’s strong three-valued logic with truth values $\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$. Semantic consequence in this logic will be denoted by \models_3 .

The fundamental technical tool is likewise the *completion* of a program:

Definition 7 Let \mathcal{P} be a normal program, consisting of clauses

$$\bar{B}^1 \wedge c_1 \rightarrow p^1(\bar{t}^1), \dots, \bar{B}^n \wedge c_n \rightarrow p^n(\bar{t}^n),$$

where p^i are atoms. The completion of \mathcal{P} , denoted by $\text{comp}(\mathcal{P})$, is computed by the following recipe:

1. choose a predicate p that occurs in the head of a clause of \mathcal{P}
2. choose a sequence of new variables \bar{x} of length the arity of p
3. replace in the i -th clause of \mathcal{P} all occurrences of a term in \bar{t}_i by a corresponding variable in \bar{x} and add the conjunct $\bar{x} = \bar{t}_i$ to the body; we thus obtain $\bar{B}^i \wedge c_i \wedge \bar{x} = \bar{t}_i \rightarrow p^i(\bar{x})$

4. for each i , let \bar{z}_i be the set of free variables in $\bar{B}^i \wedge c_i \wedge \bar{x} = \bar{t}_i$ not in \bar{x}
5. given p , let n_1, \dots, n_k enumerate the clauses in which p occurs as head
6. define $Def(p)$ to be the formula

$$\forall \bar{x}(p(\bar{x}) \leftrightarrow \exists \bar{z}_{n_1}(\bar{B}^{n_1} \wedge c_{n_1} \wedge \bar{x} = \bar{t}_{n_1}) \vee \dots \vee \exists \bar{z}_{n_k}(\bar{B}^{n_k} \wedge c_{n_k} \wedge \bar{x} = \bar{t}_{n_k})).$$

7. $comp(\mathcal{P})$ is then obtained as the formula $\bigwedge_p Def(p)$, where the conjunction ranges over predicates p occurring in the head of a clause of \mathcal{P} ¹⁰.

We see that the axioms form a normal logic program in the sense of $CLP(\mathcal{T})$, with constraints of the form $s < t$.¹¹ In the logic program, the programmed predicates are $(\neg) HoldsAt$, *Clipped* and *Declipped*. The programs for the remaining primitive predicates, i.e. the scenario and the dynamics, also form a normal logic program. We then see that applying program-completion (cf. definition 7) to a scenario in the sense of definition 4 entails that the predicates *Happens*, *Initiates* and *Terminates* are definable in terms of *HoldsAt*. While this is computationally pleasant, it involves a simplification: it may seem more reasonable to also allow *Happens* to occur in the body of 5, to allow for one event causing another given a precondition. This, however, introduces an additional loop in the computation, thus possibly affecting termination.

The fixed points of the three-valued consequence operator associated to the normal logic program then coincide with the three-valued models of the completion. The minimal fixed point is a good candidate for a minimal model in the sense discussed above. In fact, it turns out that in the cases of interest to us the consequence operator has a unique fixed point.

3.2 Structure of the minimal models

In natural language semantics it is a contested issue whether the fundamental temporal entities are points or intervals. The event calculus neatly sidesteps this issue, by taking the basic entities to be events and fluents, which are not explicit functions of time and which can be interpreted on

¹⁰By the definition of complex subgoal, \leftrightarrow does not occur in normal programs. The occurrence of \leftrightarrow in the completion is interpreted in the manner of Łukasiewicz: $\varphi \leftrightarrow \psi$ is assigned **t** when φ, ψ are assigned the same truth value in $\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$, and **f** otherwise.

¹¹It is technically convenient to replace axiom 3 by its contraposition, whose head is *Declipped*.

structures with very different ontologies for time. Even if we take the structure underlying time to be \mathbb{R} , that does not constitute an ontological commitment to points. Ontological commitment is generated rather by representation theorems, which correlate the events and fluents with point sets in a given structure. It may then very well turn out that, even when time is taken to be \mathbb{R} , fluents and events can be represented as sets of intervals, so that points have no role of their own to play.

The situation is slightly more complicated in the case of fluents admitting real parameters, for example fluents representing possibly changing partial objects. Again speaking intuitively, one would expect change to be continuous, with at most a finite number of jumps. One may thus surmise that a fluent with a real parameter should be interpreted by a semialgebraic set:

Definition 8 *A subset of \mathbb{R}^n is semialgebraic if it is a finite union of sets of the form $\{x \in \mathbb{R}^n \mid f_1 = \dots = f_k = 0, g_1 > 0, \dots, g_l > 0\}$, where f_i and g_j are polynomials.*

Intuitively at least, fluents corresponding to natural language expressions (e.g. verbs) are semialgebraic, and we would like this to fall out of the setup, without further stipulations. The kind of change it is possible to program depends on the one hand on the constraint language chosen and on the other hand on the constraint logic program. Now in *RCF* only semialgebraic sets can be defined; but would it be possible to extend the range of definable sets by a constraint logic program to *RCF*?

The main mathematical question is thus to determine what the structure of a fixed point of the consequence operator is; or, more precisely, how the scenario, the dynamics and the axioms of the event calculus jointly affect the structure of the fixed point.

The next few theorems give some pertinent results. For proofs, see [26].

Definition 9 *A finite branch in a computation tree is successful if its last node contains a constraint only. A finite branch is finitely failed if its last node is of the form $?c', G'$ (G' nonempty) with no more resolution steps possible. A query $?c, G$ is finitely evaluable if all branches in a derivation tree starting from $?c, G$ end either in success or in finite failure.*

Theorem 1 *Let *RCF* be the theory of real-closed fields. Let $\mathcal{P} = \text{EC} + \text{SCEN} + \text{DYN}$ and $?G$ a finitely evaluable query in the language of the event calculus. Then there exists a semialgebraic set c such that $\text{RCF} + \text{comp}(\mathcal{P}) \models \forall(G \leftrightarrow c)$.*

Corollary 1 *Let $\mathcal{P} = \text{EC} + \text{SCEN} + \text{DYN}$. If for all fluents $f(\bar{x})$ in the scenario and dynamics, the query $?HoldsAt(f(\bar{x}), t)$ is finitely evaluable, the theory $\text{RCF} + \text{comp}(\mathcal{P})$ has a unique model (modulo the underlying real-closed field), which is obtained after finitely many iterations of the consequence operator. In this model the primitive predicates are also represented by semialgebraic sets.*

Corollary 2 *Let $\mathcal{P} = \text{EC} + \text{SCEN} + \text{DYN}$. The following are equivalent:*

- a. *any model of $\text{RCF} + \text{comp}(\mathcal{P})$ is completely determined by its restriction to RCF ;*
- b. *$HoldsAt(f(\bar{x}), t)$ is semialgebraic.*

The hypothesis of finite evaluability of $?HoldsAt(f(\bar{x}), t)$ is rather strong. In principle, one can determine $HoldsAt(f(\bar{x}), t)$ completely by starting up derivations from both $?HoldsAt(f(\bar{x}), t)$ and $?¬HoldsAt(f(\bar{x}), t)$ and stop when the collected answer constraint in one tree is the complement of that in the other; the trees may then still contain branches which are neither successful nor failed. Let us call this notion *essential evaluability*.

Theorem 2 *Let RCF be the theory of real-closed fields. Let $\mathcal{P} = \text{EC} + \text{SCEN} + \text{DYN}$ and $?G$ a query in the language of the event calculus such that $?G$ is essentially evaluable. Then there exists a semialgebraic set c such that $\text{RCF} + \text{comp}(\mathcal{P}) \models \forall (G \leftrightarrow c)$.*

There is also a corresponding completeness result.

Theorem 3 *Let $\mathcal{P} = \text{EC} + \text{SCEN} + \text{DYN}$. The query $?HoldsAt(f(\bar{x}), t)$ is essentially evaluable if $\text{RCF} + \text{comp}(\mathcal{P}) \models \forall t (HoldsAt(f, t) \leftrightarrow c(t))$ for a constraint c .*

4 From VPs to fluents and events

In order to use the event calculus for a semantics of natural language, we need a device which codes formulas as objects or functions. We shall borrow some machinery from [10], which was applied in [11] to the semantics of nominalisation.

Let L_0 be some first order language and S_0 a theory formulated in L_0 . We briefly review the requirements on L_0 and S_0 which allow such coding.

First, one requires that L_0 contains an individual constant $\bar{0}$, a binary function symbol π and two unary function symbols π_1 and π_2 . We shall often

write (τ_1, τ_2) for $\pi(\tau_1, \tau_2)$. Secondly, we assume that S_0 proves the following statements concerning these functions

$$(11) \quad \begin{array}{ll} \text{a.} & (x, y) \neq \bar{0} \\ \text{b.} & \pi_1(x, y) = x \wedge \pi_2(x, y) = y \end{array}$$

If $M_0 \models S_0$, (\cdot, \cdot) is a pairing function in M_0 , and π_1 and π_2 are the corresponding projection functions.

One may now define tuples inductively by putting: $(\tau) = \tau$ and $(\tau_1, \dots, \tau_{k+1}) = ((\tau_1, \dots, \tau_k), \tau_{k+1})$. Similarly, one may define the corresponding projection operations π_i^k ($1 \leq i \leq k$) such that: $\pi_i^k(x_1, \dots, x_k) = x_i$.

These constructs suffice to define an abstract form of Gödel numbering.

Definition 10 *Let L be some extension of L_0 . Then we may code formulas of L as terms in L_0 . We write $\ulcorner \varphi \urcorner$ for the Gödel number in L_0 of φ in L . This notation will be used interchangeably both for the term in L_0 and for the object denoted by that term in a model M_0 .*

We will now put this machinery to work. Let φ be a formula with free variables among $x_1, \dots, x_k, y_1, \dots, y_n$. The L_0 -term $(\ulcorner \varphi \urcorner, y_1, \dots, y_n)$ contains x_1, \dots, x_k as bound variables and y_1, \dots, y_n as free variables. Since x_1, \dots, x_k are bound by abstraction, the following notation makes sense:

Definition 11 $\Delta_n \varphi[\hat{x}_1, \dots, \hat{x}_n, y_1, \dots, y_m] = (\ulcorner \varphi \urcorner, y_1, \dots, y_m)$. For $n = 1$ we will use standard set theoretical notation $\Delta_1 \{x \mid \varphi(x, y_1, \dots, y_n)\} = \varphi[\hat{x}, y_1, \dots, y_n]$. If both m and n are equal to 0, we write $\ulcorner \varphi \urcorner$.

Finally, we are able to explain what all this has to do with the event calculus. Assume verbs always have a parameter for time, as in $run(t)$. Thus, unlike event semantics in the Davidsonian tradition, we do not assume that verbs have an event parameter; rather, various kinds of eventualities will be constructed from the verb with its time parameter. Of particular importance are the *event type*¹², given formally by $\ulcorner \exists t.run(t) \urcorner$, and the *fluent*¹³ $run[\hat{t}]$. Note the difference between the two abstractions: the latter is a function of time (to truth values), while the former is an object.

¹²Also called *perfect nominal*.

¹³Also called *imperfect nominal*.

5 Eventualities

The event calculus provides a good starting point for discussing the ontology required for verb semantics. Our general picture owes much to the pioneering work of Moens and Steedman (cf. [24]), but the apparatus of the event calculus allows us to add formal precision.

An eventuality is a structure composed of four parts

1. activity (in the strict sense)
2. changing, partial object
3. culminating event
4. consequent state

This is almost like the ‘event nucleus’ of Moens and Steedman, except that we have added the ‘changing, partial object’, which plays a prominent role in the treatment of accomplishments and of the progressive. Accordingly, each VP is denoted by a quadruple, each element of which is of the form ‘-’, ‘e’ (third argument only) or ‘f’ (first, second and fourth argument)¹⁴. The elements of the quadruple are related in several ways. If the VP is represented by a structure of the form (f_1, f_2, e, f_3) , then f_1 is a fluent which corresponds to an activity, f_2 is a fluent which corresponds to a partial object which changes under the influence of f_1 (thus f_2 will in general contain variables), e is both the culminating event of the activity and an initiating event for the resulting consequent state, f_3 . The structure thus satisfies the properties $Initiates(e, f_3, t)$ and $Terminates(e, f_1, t)$. There is no distinguished role for an event initiating f_1 , since there may be many of them, for example if the activity f_1 is interrupted from time to time. It is of course implicit that the activity f_1 is nontrivial, so that the scenario must contain a reference to an initiating event, perhaps in the form *Initially*.

The fluents in this structure play different roles in a scenario and in the dynamics. f_1 is not allowed to occur in *Releases* and in the third argument of *Trajectory*, but may occur in the latter’s first argument. This is because activities are volitional and must hence be initiated and terminated by actions.

¹⁴For simplicity we have treated only the case where the direct *object* is partial and changing. A similar phenomenon can occur in subject position however, as in the well-known causative/inchoative alternation

- (i) a. The cook thinned the gravy.
b. The gravy thinned.

For f_2 it is the other way around; since states are causally inert, they are not allowed as the first argument in *Trajectory*. Usually, the consequent state f_3 will be an instantiation of f_2 , so it inherits the latter’s syntactic restrictions. The relationship between the fluents f_1 and f_2 is given by the dynamics, i.e. a statement of the form $S(f_1, f_2, t, d) \rightarrow \text{Trajectory}(f_1, t, f_2, d)$, where $S(f_1, f_2, t, d)$ is a state in the sense of definition 3.

Note that an eventuality *determines* the scenario and dynamics in the sense that these should at least fix the meaning of the positive components of the quadruple. Thus, for a quadruple of the form $(f_1, f_2, -, -)$ nothing needs to be said about a culminating event, but this changes when the remaining slots are filled, for example when an activity (*draw*) is coerced into an accomplishment (*draw a circle*). Often the scenario will need to contain more, for example a reference to an initiating event. As will be seen in the following sections, different *Aktionsarten* correspond to the various ways in which the slots in the quadruple can be filled.

5.1 Formal definition of aspectual classes

We will define aspectual classes as specific types of eventualities, not as verb classes, because we believe, as mentioned in the introduction, that verbs can by and large be coerced into any aspectual class. As we have seen an eventuality can contain an event and up to three fluents; one may then distinguish types of eventualities according to the slots filled in the quadruple.

State	(-, -, -, +)
Activity (strict)	(+, -, -, -)
Activity (wide)	(+, +, -, -)
Accomplishment	(+, +, +, +)
Achievement	(-, -, +, +)
Points	(-, -, +, -)

We will add a few comments on this table.

1. - indicates that a slot is empty, + that it is filled with an object of the appropriate category.
2. The distinction between states and activities is given by the different syntactic roles they play in the event calculus. The following table shows the argument slots that can be filled by states and activities (Traj_i refers to the i^{th} argument of *Trajectory*)

	<i>Releases</i>	<i>Traj.1</i>	<i>Traj.3</i>
States	+	-	+
Activity (strict)	-	+	-

There are no restrictions on occurrences in the other primitive predicates.

3. The difference between an activity in the strict sense (cf. *push*) and in the wide sense (*push a cart*), is that in case of the latter what Dowty [8] calls the ‘incremental theme’ has been added not the cart itself, but the changing position of the cart.
4. The quadruples occurring in the above list can all be described by VPs. It is an interesting question whether some of the other quadruples also characterise VPs. Observe that there are quadruples which characterise eventualities corresponding to *sentences*. Here are a few examples. To this end, it is useful to give a wider interpretation to the second component, which we may also take to denote a gradable state, such as “love to a certain degree”. With this stipulation, a quadruple like $(-, +, -, +)$ is exemplified by the sentence

(12) For a while she remembered him dimly, but she soon forgot him completely.

In the same vein, the quadruple $(-, +, +, +)$ corresponds to

(13) She felt some affection for him for a while, but after that incident, her love was dead.

In these examples we have used quasi-Boolean operations such as ‘but’, the question is whether there are simple VPs which need to be characterised by such quadruples.

5.2 Example of an accomplishment: ‘build a house’

To fix ideas, we give a scenario for the accomplishment ‘build a house’, thus illustrating the role of the various components in the quadruple. An accomplishment is characterised by a quadruple $(+, +, +, +)$, so that the scenario to be written must contain a reference to all these components.

With this in mind, we need the following terms in the language of the event calculus.

1. *build* is an activity fluent
2. *house(x)* is a parametrised fluent representing the construction stage (*x*) of a house
3. *c* is a real constant indicating a construction stage at which the house is considered finished; thus *house(c)* is the fluent representing the consequent state.
4. *start* is any event initiating building.
5. *finish* is the canonical event terminating building, namely when the house is finished.
6. *g* is a real-valued function relating the building activity to the construction stage.

Note that so far we have not introduced an object *house*; all we have is a fluent *house(c)*. However, nothing prevents us from stipulating that $House(house(c))$, thus effectively turning *house(c)* into an object as well.

1. $Initiates(start, build, t)$
2. $Initiates(finish, house(c), t)$
3. $Terminates(finish, build, t)$
4. $HoldsAt(build, t) \wedge HoldsAt(house(c), t) \rightarrow Happens(finish, t)$
5. $Releases(start, house(x), t)$
6. $HoldsAt(house(g(t)), t) \rightarrow Trajectory(build, t, house(g(t + d)), d)$

The last two statements, 5 and 6, are jointly called the *dynamics*; this is characteristic of both activities (in the wide sense) and accomplishments. Accomplishments are distinguished from activities (in the wide sense) by statements describing the behaviour of the canonical terminating event, here statements 2 – 4.

The preceding statements should be thought as being part of the lexical entry of ‘build a house’; of course, the scenario for ‘build’ itself is much more complex than that presented here. In the context of a sentence, episodic information will be added, such as

7. $Happens(start, t_0)$

8. *Initially(house(a))*,

where a is a constant representing a contextually determined initial construction stage of the house.

In [11] we have explained at length how verbs and fluents are related. Briefly, the fluent *build* results from the transitive verb x *builds* y by existentially quantifying the object position (y) and then nominalising. One may furthermore view the participle *built* as arising from *build* by existentially quantifying the subject position (x). The consequent state then satisfies $\exists x \textit{builds}(f(c))$; thus there is a pleasant symmetry between the first and last component of the quadruple.

5.3 Example of an achievement: ‘reach the top’

Here we need a terminating event¹⁵ *reach* and a fluent *be-at-the-top*, related by

1. *Initiates(reach, be-at-the-top, t)*.

Episodic information can now only be of the form

2. *Happens(reach, t₀)*

Note that negation as failure will ensure that *Initially(be-at-the-top)* will be false.

5.4 The progressive

The following sentence makes perfect sense

- (14) John was building a house last year, but he had to stop when his credit failed.

Upon reflecting, there is something paradoxical about this sentence. Whereas it belongs to the meaning of the accomplishment *build a house* that the activity (‘build’) is directed toward the consequent state of a finished (‘built’) house, the actual occurrence of that consequent state can be denied without contradiction. So how can a seemingly essential component of the meaning be denied, without affecting the meaning itself? This is known as the ‘imperfective paradox’. The literature is replete with attempted resolutions of the paradox, ranging from explaining the problem away (cf. the recent Michaelis [14]) to various invocations of possible worlds (see Dowty [7], Landman [13] or de Swart [5]). Possible worlds solutions are based upon the idea that

¹⁵‘Terminating’ with respect to an activity not explicitly mentioned.

The progressive picks out a stage of a process/event which, if it does not continue in the real world, has a reasonable chance of continuing in some other possible world [5, p. 355].

but differ in the (largely informal) descriptions of the possible worlds used. For example, [7] uses so-called ‘inertia worlds’, worlds which are identical with the present world until ‘now’, but then continue in a way most compatible with the history of the world until ‘now’. Thus these approaches are intensional in the formal sense of using possible worlds. In fact, most authors (but not all) would agree that the progressive creates an intensional context: even though John may have stopped building at a stage when it was unclear whether he was building a house or a barn, still only one of

- (15) a. John was building a house.
b. John was building a barn.

can be true of the situation described.

Our solution (first proposed in [11]) will use the event calculus, but before we go into this, we discuss Michaelis’ attempt in [14] to explain the problem away; this will then show why an elaborate machinery is necessary.

Explicitly denying that the progressive creates an intensional context, Michaelis writes

Under the present proposal, the Progressive sentence *She is drawing a circle* denotes a state which is a subpart not of the accomplishment type *She- draw a circle* but of the activity type which is entailed by the causal representation of the accomplishment type. Since this activity can be identified with the preparatory activity that circle drawing entails, circle drawing can in principle be distinguished from square drawing etc. within the narrow window afforded by the Progressive construal [and] does not require access to culmination points either in this world or a possible world . . . [14, p. 38].

We find this rather doubtful. Without access to a person’s intention it may be very hard to tell initially whether she is drawing a circle or a square, building a barn or building a house. But that person’s intention in performing a activity is characterised precisely by the associated consequent state, even though the latter cannot yet be inferred from the available data.

Here the event calculus comes to our rescue. In the event calculus, an activity comes with a theory (the scenario) which describes the intended

consequences of that activity; this is precisely the function of the dynamics. However, unlike approaches such as Parsons' [17], where one quantifies existentially over events, the scenario is a universal theory and does not posit the occurrence of the intended consequences. Their existence is guaranteed only in minimal models of the scenario combined with the axioms for the event calculus, in which no unforeseen obstacles occur. Thus, the meaning of an accomplishment (as embodied in the scenario) involves a culminating event *type* (which therefore must exist); but there are no existential claims about the corresponding event *token*¹⁶. These are handled by different mechanisms. The use of the progressive indicates that the appropriate mechanism is default inference. By contrast, tense can be defined via ordinary existential quantification, so stays within the realm of monotonic inference.

We will now make the above considerations slightly more formal. Consider the sentence

(16) John is building a house.

Viewed purely denotationally, this is a statement about an activity, formally $HoldsAt(build, now)$, conjoined with the scenario describing the lexical meaning of 'build a house'. We claim however, that the progressive also involves a particular way of looking at the culminating event: that its occurrence is to be expected. We will see presently that the set of statements associated to (16) nonmonotonically implies that the house will be finished. If we apply the progressive not to an accomplishment but to an activity, there is no canonical culminating event. But the crucial presupposition for the use of the progressive remains the presence of a dynamics.

Theorem 4 *Let \mathcal{P} be the constraint logic program (w.r.t. a real-closed field) consisting of EC, the scenario 1–6, and the episodic statements*

7. $HoldsAt(build, now)$.

8. $Initially(house(a))$.

Suppose $\lim_{t \rightarrow \infty} g(t) \geq c$. Then $comp(\mathcal{P})$ has a unique model, and in this model there is a time $t \geq now$ for which $HoldsAt(house(c), t)$. By virtue of the stipulation that $House(house(c))$, there will be a house at time t .

The effect of taking the completion is that in a model, only those events occur which are forced to happen due to the scenario; similarly, only those

¹⁶And similarly for the state consequent upon a culminating event type.

influences of events are considered which are explicitly stated in the scenario. In this particular case, since the scenario does not mention the event ‘going broke’, the completion excludes this possibility; and similarly for other possible impediments to completing the construction. It is then a consequence of the general result theorem 1 that the completion actually has a unique model.

Notice, however, that the existence of a time at which the house is finished is only guaranteed in minimal models. Thus, this inference is non-monotonic: if we obtain more episodic information, the conclusion may fail.

6 Explaining coercion

The default eventuality associated to, say, ‘reach the top’, is an achievement. As such, the application of the progressive is excluded, because the achievement lacks the requisite dynamics. Nevertheless, in some contexts the progressive is appropriate, as in

(17) They were reaching the top when a blizzard forced them to go back.

In this case the achievement ‘reach the top’ is *coerced* into an accomplishment, to which the progressive can be applied felicitously. The purpose of this section is to give a formal treatment of this operation, or rather, family of operations. We will distinguish three forms: additive coercion, subtractive coercion, and cross-coercion.

6.1 Additive coercion

This is the simplest kind of coercion, which consists in elaborating a scenario.

Activity \rightsquigarrow accomplishment The verb ‘build’ is an activity, which is transformed into an accomplishment by adding a direct object such as ‘a house’. In the simple set up outlined above, the only property governing ‘build’ is 1. The effect of adding the direct object is that this scenario is extended with a dynamics, and a set of statements describing the behaviour of the culminating event. Hence the name ‘additive coercion’.

Achievements \rightsquigarrow accomplishments Above we remarked that sentence (17) requires the transformation of ‘reach the top’ into an accomplishment. That is, since the progressive requires for its application the presence of a dynamics, this must first be added to the scenario. This can be done

by adding an activity *climb*, a changing partial object *height(x)* representing the height gained during climbing, a dynamics, and a statement relating the event *reach* to the activity, say by means of $HoldsAt(climb,t) \wedge HoldsAt(height(8.850m),t) \rightarrow Happens(reach,t)$.

6.2 Subtractive coercion

We now discuss several instances of the converse transformation, in which parts of a scenario are deleted.

Accomplishment \rightsquigarrow activity The most interesting case of this form of coercion occurs when we consider bare plurals: ‘drink a glass of wine’ is an accomplishment, but ‘drink wine’ is an activity (in the wide sense). What the activity and the accomplishment have in common is a dynamics, the crucial sentence of which has the following form

1. $HoldsAt(f_2(g(t)),t) \rightarrow Trajectory(drink,t,f_2(g(t+d)),d)$.

Here, f_1 is the activity, and f_2 is the changing partial object. Now activities and accomplishments are differentiated from each other by the fact that the scenario for an accomplishment requires sentences of the type

2. $Terminates(e(c),f_1,t)$
3. $HoldsAt(f_2(c),t) \wedge HoldsAt(f_1,t) \rightarrow Happens(e(c),t)$.

Here, $e(c)$ is a culminating event, dependent upon a constant c . An example of such a constant would be the quantity determined by a glass of wine; then $e(c)$ is the event ‘finish a glass of wine’. Looking at the last sentence, one sees that, in a minimal model, the culminating event will only be activated if for some t , $g(t) = c$ can be derived.

Thus, the effect of ‘a glass of –’ would be to add 2. and 3. Conversely, taking away ‘a glass of –’ from ‘drink a glass of wine’ would lead to the deletion of sentences 2. and 3.

6.3 Cross-coercion

We now discuss several difficult cases of coercion which cannot be viewed as adding or deleting sentences from a scenario. This kind of coercion occurs for instance when a state (which is of the form $(-, -, -, +)$) is coerced into an activity (of the form $(+, +, -, -)$) under the influence of the progressive. Of course a combination of the addition and subtraction operations would also yield this transformation, but presumably something else is going on.

State \rightsquigarrow **activity** Croft [4] gives the following convincing examples:

(18) She is resembling her mother more and more every day.

(19) I am loving her more and more, the more I get to know her.

Consider example (18). Let f be a fluent denoting *resemble*, conceived of as a state. Then the sentence

(20) She resembles her mother.

is represented as $HoldsAt(f, t)$.

The phrase ‘more and more’ introduces a new fluent $f'(x)$ denoting *resemblance to degree x* . Furthermore, the application of the progressive introduces a dynamics consisting of the general form

1. $Initiates(e, f_1, t)$
2. $Releases(e, f_2, t)$
3. $HoldsAt(f_2, t) \rightarrow Trajectory(f_1, t, f_2, d)$.

Here, the f_1, f_2, e are *variables*, which have to be unified with terms provided by the discourse, namely f and $f'(x)$, and a contextually determined starting event e_0 which satisfies

1. $Happens(e_0, t_0)$
2. $t_0 < now$.

We claim that the observed coercion is due to the unification of f with the variable f_1 . Suppose that f is substituted for f_2 in $Releases$ and $Trajectory$ predicate, then $Clipped(s, f, r)$ will be true of f if $s < t_0 < r$). This means that the query $?\neg Clipped(s, f, r)$ will fail for these s, r . By contrast if f is substituted for the first argument of the $Trajectory$ predicate then $f'(x)$ can be substituted in the third argument and in $Releases$ and successful computations can be constructed as usual. This implies that the state *resemble* is coerced into an activity via unification.

It should be noted that some native speakers of English find this example more problematic than Croft appears to do. From the formal side we may observe that this example involves much more processing than all the previous examples; it is possible that this difference is related to the difference in acceptability judgments.

The same analysis can then be applied to

(21) I am loving her more and more every day.

Examples (21) and (19) are distinguished by the fact that for (19) there are two candidate activity fluents, *love* and *get to know her* and accordingly two possible unifications. The case where ‘love’ is coerced to be the activity is similar to example (18). When *get to know her* is interpreted to be the activity, this means that a new changing partial object is introduced, namely the changing degree of knowledge $g'(x)$ which is related to the degree of loving $f'(x)$ by means of some real-valued function h as in formula (22).

$$(22) \quad \forall x \forall t (HoldsAt(g'(x), t) \rightarrow HoldsAt(f'(h(x)), t)).$$

As before the progressive introduces a dynamics. Let the fluent g represent *get to know her*. By an argument similar to the one above, g is unified with the first variable of the *Trajectory*-predicate and $g'(x)$ with the third variable. This then yields a prediction for $f'(x)$ by means of the formula (22).

Points \rightsquigarrow **activity** Here we must consider a transformation from a quadruple $(-, -, +, -)$ to $(+, -, -, -)$ (or perhaps $(+, +, -, -)$). Consider

- (23) a. The light flashed.
 b. The light was flashing all night.

‘Flash’ is a point; for this example we will therefore consider only the event *flash*. Sentence (23-a) can then be formalised as

1. $Happens(flash, t_0)$
2. $t_0 \leq now$

In sentence (23-b) ‘flash’ has become an activity, hence formally a fluent, or pair of fluents. In [11] we have indicated (for the case of coerced nominals) a procedure which achieves just this. Namely, coercion is represented by the mapping $flash \mapsto Happens[flash, \hat{t}]$. The latter expression is a fluent. We propose that this fluent is identified with the changing partial object, and that an unspecified activity f_1 is added to which $Happens[flash, \hat{t}]$ is related via a dynamics. The only constraint this f_1 has to satisfy is that its corresponding finite set of intervals and/or points (which exists due to theorem 1) coincides with that of $Happens[flash, \hat{t}]$. The dynamics may then simply be defined by $HoldsAt(Happens[flash, \hat{t}], t) \rightarrow Trajectory(f_1, t, Happens[flash, \hat{t}], d)$.

A non-instance: the progressive The progressive is often described as a stativizer, so that the main clause in (24-a) below would denote a state. The basis of this judgment is a comparison of (24-a) to (24-b) and (24-c):

- (24)
- a. He was crossing the street when the truck hit him.
 - b. He was asleep when his friend came in.
 - c. He got up when his friend came in.
 - d. John drank wine when his son came home from school. John drank whisky when his wife returned from work. John's family thought he must be an alcoholic.

Sentence (24-a) is like (24-b), which has a stative main clause, in that the event mentioned in the subordinate clause takes place within the interval corresponding to the main clause. This is in contrast to (24-c), where the main clause denotes an event which takes place after the event denoted by the subordinate clause.

This argument does not suffice to classify the progressive as an operator which yields states as values. The main clause in (24-a) still refers to a period in which change is taking place, unlike the main clause of (24-b). Even without the progressive, the activities in (24-d) can be interpreted as in (24-b)¹⁷, and one would not want to say that *they* are stativized. The main clauses of (24-a), (24-b) and (24-d) have in common that they refer to intervals, unlike (24-c). Furthermore, the meaning of *when* is sensitive to the temporal profile of its first argument; more on this below. But it should be clear by now that temporal profile is only a small, derivative part of the meaning of a verb.

Activity/accomplishment \rightsquigarrow state To conclude the section on coercion we briefly discuss two examples about which much more should be said, in particular with regard to the syntax-semantics interface.

Negation It has often been held (see for example [27]) that negating an activity results in a stative predicate. This intuition can to a certain extent be reproduced in the present framework. An activity (in the wide sense) consists of a fluent representing the activity proper, and a changing partial object. In a minimal model, the partial object changes only when the activity is 'on'; outside of those intervals, the partial object does not change. It follows that the set of intervals complementary to that representing the activity has a state-like character. It differs from a true state in our sense

¹⁷There is also an event reading comparable to (24-c).

by being initiated and terminated by events, since of course a terminating event for an activity is an initiating event for its negation.

Passive In German and Dutch there is a form of the passive, the ‘Zustandspassiv’ (indicated by a special auxiliary), which transforms an activity or an accomplishment into a (consequent) state.

- (25) a. Johann baut ein Haus. Es ist weiss.
b. Das Haus ist von Johann gebaut. Es ist weiss.
c. Das Haus wird von Johann gebaut. *Es ist weiss.

In (25-b), the auxiliary *sein* indicates that the house is finished; by contrast, in (25-c) the auxiliary *werden* refers to the process of building. This explains the * in (25-c).

In English the passive is apparently ambiguous between the two readings. Compare

- (26) a. John is building a shed in his garden. This causes his neighbours much distress [because of the tremendous noise].
b. The shed that is built in John’s garden causes his neighbours much distress [because it spoils their view].
c. A shed is built in John’s garden. This causes his neighbours much distress.

‘This’ in sentence (26-c) can refer both to an activity, as in (26-a), and the result of that activity, as in (26-b). If one conceives of the passive syntactically as movement of the object NP into subject position, an interesting tension between syntax and semantics comes to the fore. The NP *a house* in *build a house* is an incremental theme, and need not denote an object in the ontological sense. In our set up, the denotation of *a house* is as it were distributed over the changing partial object, the canonical terminating event, and the consequent state whose relations are governed by the scenario. Similarly, the verb *build* is not a two-place predicate, but a fluent with one (subject) parameter. So what happens semantically when the NP is moved into subject position?

One possibility is that nothing happens, in which case a passive sentence retains the process reading of the corresponding active sentence. Another possibility is that the NP is re-interpreted as a real object, to which an adjective can be applied. Indeed, in English grammar this form of the passive is known as ‘adjectival passive’. For example, *built* would now be an adjective, obtained from the verb *build* by existentially quantifying the subject

position. But the upshot of this is that ‘a house is built’ now corresponds to the consequent state of the accomplishment, as indicated in section 5.2.

6.4 Tense

The main purpose of this section is to state a formal version of the following intuitive idea. The semantic force of the progressive as applied to accomplishments is the introduction of nonmonotone reasoning. The effect of the past tense is to enforce a monotone conclusion; that is, from *John built a house* we want to derive in a monotonic way that a house exists. Of course we do not claim that this short section contains a comprehensive theory of tense.

We suggest definitions for the past and the futurate tense as given in 13. This, however, does not mean that we think these are the only tenses which have to be distinguished in natural languages¹⁸. For example, it is assumed that ChiBemba has a much more elaborate system of past and future tense than English (compare [16], p. 138). But at least past seems to be one of the most basic tenses in the sense described below. In [23], Stassen argues on the basis of a sample containing about 400 languages for what he calls the *Tensedness Parameter*:

Definition 12 [*Definition of the Tensedness Parameter*]

- a. *If a language has a grammatical category of tense which*
 - (1) *is morphologically bound on verbs, and*
 - (2) *minimally involves a distinction between past and nonpast time reference, then that language is tensed.*
- b. *In all other cases, a language is non-tensed.*
[23], p. 350/51

The distinction between past and nonpast is, according to Stassen, the defining typological criterion for assuming that a language has tense. Since past seems to be the most typical temporal operator, we will concentrate in the following exclusively on this tense.

Definition 13

$$Past(\phi[\hat{s}]) \Leftrightarrow \exists e \exists t (Happens(e, t) \wedge Terminates(e, \phi[\hat{s}], t) \wedge t < now)$$

$$Fut(\phi[\hat{s}]) \Leftrightarrow \exists e \exists t (Happens(e, t) \wedge Initiates(e, \phi[\hat{s}], t) \wedge t > now)$$

¹⁸For a recent overview paper on tense see [2].

We exemplify the above definition by first applying it to a simple activity sentence. *John ran* is translated as: $Past(run(j, s))$. This is true iff

$$\exists e \exists t (Happens(e, t) \wedge Terminates(e, run[j, \hat{s}], t) \wedge t < now).$$

Since the fluent $run[j, \hat{s}]$ is true at the terminating time t , we get $HoldsAt(run[j, \hat{s}], t)$, from which we can derive¹⁹ $\exists t(run(j, t) \wedge t < now)$.

If we translate *John was running* as $HoldsAt(run[j, \hat{s}], t) \wedge t < now$ ²⁰, the above calculation shows that *John ran* implies *John was running* but not vice versa because there may not be a terminating event for the running activity. This result is empirically supported by the following pair of sentences:

- (27) a. John was running and he still does.
 b. *John ran and he still does.

For the most difficult aspectual class – accomplishments – the effect of past is slightly different. It says that the nonmonotonic inference engendered by the progressive is now monotonic. To be more precise, consider again the scenario for building a house in section 5.2. *John built a house* says that the canonical event *finish*, which terminates building when the house is finished, happened some time before *now*. This means that in the *Past*-scheme we don't have quantification over events and times but only over times. This results in:

$$\exists t (Happens(finish, t) \wedge Terminates(finish, build, t) \wedge t < now).$$

Moreover, the lexical semantics of *build a house* requires *finish* to be an initiating event too. From clause 2 of the scenario we get: $Initiates(finish, house(c), t)$. From this follows that for time s after t , $HoldsAt(house(c), s)$ is true. Given the stipulation $House(house(c))$, we get that there is a house at a time s before *now*.

¹⁹This is true because $HoldsAt$ is a truth predicate; i.e. the following equivalence holds:

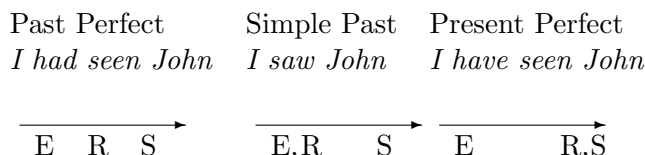
$$HoldsAt(\phi[\hat{s}], t) \leftrightarrow \phi(t)$$

See [11] for a detailed exposition of the combination of the event calculus with a theory of truth.

²⁰This translation was proposed in [11]. We think that tense as applied to sentences in the progressive should be analysed differently from tense as applied to their non-progressivised counterparts. The first type of sentences is analysed with the help of the $HoldsAt$ predicate, the second via the $Happens$ predicate.

Finally, we will at least indicate how Reichenbach's (see [20]) more elaborate treatment of tense can be captured in the system we propose. Reichenbach distinguishes between speech time, event time and reference time. His speech time is simply our *now*, and the event time can be both the time where a fluent holds or the time at which an event occurs. Let us turn to Reichenbach's notion *reference time* which is the new ingredient of his approach.

Reichenbach thought of the reference time as the time we are talking about. The following picture from [24] explains his intuition. S, E, R are short for speech time, event time and reference time and the arrow indicates the flow of time.



The interesting example is the past perfect. Here we have to distinguish all three times. The event expressed by the sentence is presented by the past perfect as prior to a time in the past. The event time is presented from the point of view of a past reference time. In the other examples, the reference time is coincidental with either the event time (simple past) or the speech time (present perfect). According to Steedman [24] this difference is responsible for the grammaticality distribution in (28).

- (28) a. *I have lost my watch but I have found it again.
 b. I lost my watch but I (have) found it again.

In many examples reference time is introduced implicitly, but in the sentences (29) the subordinate clause mentions the reference time explicitly²¹.

- (29) a. When John entered, Bill was asleep. (state)
 b. When John entered, Bill was shaving. (activity)
 c. When John entered, Bill left. (achievement)
 d. Archimedes was drawing a circle when a Roman soldier killed him. (accomplishments)

In (29-a), (29-b) and (29-d) the time introduced by the *when*-clause overlaps

²¹We cannot discuss Reichenbach's suggestion and its development in modern semantics in detail here. The interested reader is referred to [12] and [24] for a more elaborate exposition.

the time of the main clause. In (29-c) however, Bill is absent *after* the entering of John.

Standardly it is assumed that the *when*-clause introduces the reference time in Reichenbach's sense which then serves as a temporal anchor for the time of the main clause.

The examples in (29) will be represented as pairs (e, e') or (e, f) , where e represents the event introduced by the *when*-clause and e' or f the *event* or *fluent* expressed by the main clause, depending on the aspectual class of the main clause.

We propose the following semantic principle:

- (30) The event introduced by the *when*-clause is always related to the first positive component of the quadruple expressed by the main clause.

Therefore, we get *fluents* as second components for states, activities and accomplishments and an *event* for achievements. To keep notation simple, we write *enter* for the event of *John's entering* and *shaving* for the fluent representing *Bill's shaving*. Sentence (31-a) is then formalised as in (31-b).

- (31) a. When John entered, Bill was shaving.
 b. $\exists t(Happen(enter, t) \wedge HoldsAt(shaving, t)) \wedge t < now$

Since *shaving* is an activity and activity fluents are not allowed as arguments of the *Releases*-predicate, it follows from corollary 2 of theorem 1 that the following proposition is true:

$$HoldsAt(shaving, t) \leftrightarrow \exists s, r(s < t < r \wedge \forall t' \in (r, s) HoldsAt(f, t'))$$

Therefore, we get an overlap between the reference time and the time for which the fluent *shaving* holds. A similar analysis applies to accomplishments, since the first positive component of the quadruple representing an accomplishment is again an activity fluent. As a result, we get that the Roman soldier killed Archimedes during the time when the circle-drawing fluent held.

The explanation for achievements is slightly different. The fluent *absence* represents the second positive component of the achievement *leave*, i.e. the result fluent which is initiated by *John's leaving*.

- (32) a. When John entered, Bill left.
 b. $\exists t(Happen(enter, t) \wedge Happens(leave, t) \wedge Initiates(leave, absence, t)) \wedge t < now$

Since $HoldsAt(absence, t)$ is not true for the time t at which the event *leave* initiates the fluent *absence* but immediately after t , we get that the absence of Bill holds after the entering of John.

7 Conclusion

We can now explain in more detail why coercion is considered an intensional phenomenon in this paper.

The following formal reconstruction of Frege’s notion *sense* is from [26].

Definition 14 *The sense of an expression is the constraint logic program (in the sense of $CLP(T)$) representing the lexical component, viewed as an algorithm which transforms an episode into the denotation of the expression in a model, using the axioms of the event calculus.*

Consider again the explanation for the coercion effect which turns the stative sentence *She resembles her mother* into the activity expressed by *She resembles her mother more and more every day*.

The explanation is based on two possible unifications for fluent variables in the *Releases* and *Trajectory* predicates of the enriched scenario. However under the first substitution successful computation is blocked because $Clipped(s, f, r)$ is derivable under this unification. This failure is reached in a *finite* number of steps. By contrast the other possible substitution leads to a successful computation of a minimal model.

We can thus describe *coercion* as a map from the sense of an expression to a different but related sense of an expression. The relationship is determined both by linguistic and non-linguistic factors depending on the type of coercion involved.

The last section on *tense* is meant to indicate at least how tense is related to sentences in the progressive versus their non-progressivised counterparts. Sentences not in the progressive tense are subject to the *Happens* predicate of the event calculus, while sentences in the progressive are subject to the *HoldsAt* predicate.

Tense is therefore not considered as a uniform phenomenon in this paper. However, this strategy explains the role monotone versus non-monotone reasoning plays for the sentence types under consideration.

References

- [1] E. Bach. The algebra of events. *Linguistics and Philosophy*, 9:5–16, 1986.
- [2] R. Bäuerle. Issues in the semantics of tense. In F. Hamm and E. Zimmermann, editors, *Linguistische Berichte, Sonderheft: SEMANTIK*. Buske Verlag, Hamburg, 2002.
- [3] G. Bealer and U. Mönnich. Property theories. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic Vol. IV*. Reidel, Dordrecht, 1989.
- [4] W. Croft. The structure of events. In M. Tomasello, editor, *The new psychology of language*. Lawrence Erlbaum Associates, Mahwah, NJ, 1998.
- [5] H. de Swart. Aspect shift and coercion. *Natural Language and Linguistic Theory*, 16:347–385, 1998.
- [6] J. Dölling. Polysemy and sort coercion in semantic representation. In P. Bosch and P. Gerstl, editors, *Discourse and lexical meaning*. Arbeitspapiere des SFB 340, Nr. 30, Stuttgart/Tübingen, 1993.
- [7] D. Dowty. *Word Meaning and Montague Grammar*. Reidel, Dordrecht, 1979.
- [8] D. Dowty. Thematic proto-roles and argument selection. *Language*, 67:547–619, 1991.
- [9] M. Egg. Flexible semantic construction: the case of reinterpretation. Habilitationsschrift, 2000.
- [10] S. Feferman. Toward useful type-free theories i. *The Journal of Symbolic Logic*, 49:75–111, 1984.
- [11] F. Hamm and M. van Lambalgen. Event calculus, nominalisation and the progressive. Research report, ILLC, Amsterdam, December 2000. 76 pp. to appear in *Linguistics and Philosophy*. Available at <http://www.semanticsarchive.net>.
- [12] H. Kamp and U. Reyle. *From Discourse to Logic*. Reidel, Dordrecht, 1993.

- [13] F. Landman. The progressive. *Natural Language Semantics*, 1:1–32, 1992.
- [14] L. Michaelis. A unification-based model of aspectual type-shifting. University of Colorado at Boulder, 2001.
- [15] M. Moens and M. Steedman. Temporal ontology and temporal reference. *Comput. Ling.*, 14:15–28, 1988.
- [16] W. O’Grady, M. Dobrovolsky, and M. Aronoff. *Contemporary Linguistics: An Introduction*. St. Martin’s Press, New York, 1991.
- [17] T. Parsons. *Events in the Semantics of English*. MIT University Press, Cambridge MA, 1990.
- [18] S. Pulman. Aspectual shift as type coercion. *Transaction of the Philological Society*, 95:279–317, 1997.
- [19] J. Pustejovsky. Type coercion and lexical selection. In J. Pustejovsky, editor, *Semantics and the lexicon*. Kluwer, Dordrecht, 1993.
- [20] H. Reichenbach. *Elements of Symbolic Logic*. Dover Publications, Inc., New York, 1947.
- [21] J. Saul. Substitution and simple sentences. *Analysis*, 57:114–118, 1997.
- [22] M.P. Shanahan. *Solving the frame problem*. The M.I.T. Press, Cambridge MA, 1997.
- [23] L. Stassen. *Intransitive Predication*. Clarendon Press, Oxford, 1997.
- [24] M. Steedman. Temporality. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*. Elsevier, Amsterdam, 1997.
- [25] P.J. Stuckey. Negation and constraint logic programming. *Information and Computation*, 118:12–33, 1995.
- [26] M. van Lambalgen and F. Hamm. Moschovakis’ notion of meaning as applied to linguistics. In M. Baaz and J. Krajčec, editors, *Logic Colloquium ‘01*. A. K. Peters, Natick, Mass., 2002.
- [27] H. Verkuyl. *A Theory of Aspectuality. The interaction between Temporal and Atemporal Structure*. Cambridge University Press, Cambridge, 1993.

Fritz Hamm
Seminar für Sprachwissenschaft
der Universität Tübingen
Wilhelmstr. 113
72074 Tübingen
e-mail: friedrich.hamm@uni-tuebingen.de

Michiel van Lambalgen
FGW / ILLC
Universiteit van Amsterdam
Nieuwe Doelenstraat 15
1012 CP AMSTERDAM
e-mail: vanlambalgen@hum.uva.nl