Content in Context

Alice G.B. ter Meulen
Department of Philosophy SY 026
Indiana University
Bloomington IN 47405 USA
atm@phil.indiana.edu

Abstract

The way in which we express information in time about time in ordinary English provides a core empirical domain to study the dynamics of interpretive and inferential processes in context. This paper advocates the view that contexts are tree-structures that represent static and dynamic information content, to allow for a configurational characterization of situated inference. In an interpretation of a sequence of sentences we obtain descriptive, aspectual and perspectival information. Finite labelled trees, called Dynamic Aspect Trees (DATs), with two node sorts represent these three kinds of information in an integrated way, leading to a new notion of a chronoscope, the locally consistent information-stage. DATs are interpreted in event-structures which satisfy certain domain specific semantic constraints based on inferential properties of progressive, perfect and past tense inflections. Aspectual information controls how such DATs are updated, which may or may not affect the given chronoscope. Static information is represented by stickers that are portable into new chronoscopes given specific thresholds, configurationally defined in the DATs. The notion of situated inference models the context-dependent reasoning about the flow of time.

1. Introduction

When we describe something that happened, we make, quite unreflectively, certain choices in inflecting the verbs for tense and aspect. Eventhough we never pay much attention to it, the inflection we chose contributes information used in our reasoning about the described episode. To characterize this reasoning in context using tense and aspect, we should model not only how the temporal information given by a text is dynamically represented, but also how further information may be extracted at any point from such representations by inferential processes. An account of temporal reasoning needs to formulate under what conditions a conclusion may be derived from a given representation with a progressive, a perfect or a simple past verbal inflection. Such inferential rules depend not only on the inflection with which the information was initially given, but also on the order in which it was presented in the text and what other information intervened. Whatever form these contexts take, they should be suitable for defining such ‘situated’ inferences in terms of their configurational properties. This paper proposes that finite labelled trees constitute a semantically clear and flexible tool for such representations of temporal information, integrating the descriptive, aspectual and perspectival information used in reasoning. Reasoning about change and time is inherently context-dependent, as the flow of information is simultaneously in time as well as about time. Such reasoning about time is dependent upon specific syntactic and semantic properties of the linguistic form in which the information is presented to us. In particular, the temporal order in which the information is given often (but not always) matters significantly and the aspectual class of the description contributes essential control information. The classification of the descriptive information contained in an inflected clause into aspectual classes is a complex process. It is partly lexically driven, partly by the argument-structure of the main verb, but also depends on the presuppositions of both the locally given and the new information and, ultimately also on core logical relationships as well as upon our common sense knowledge of the world. Quantificational, conditional or generic information and all negative information is represented in a sticker, as static information that does not change the context, but constrains the set of possible dynamic updates constructing new chronoscopes. The representation of the temporal information should depend on all such parameters in the context. In this paper we assume the mapping from English surface structure into aspectual classes as given (see ter Meulen 1995).

Various linguistic forms may be used to give stative information: statements with progressive and perfect inflection, those with main verbs be or have , lexical statives (sit, reside), attitude verbs, and those containing generic NPs referring to kinds. All information given with an internally negated predicate is also represented as sticker. Only some of these forms are sensitive to the textual order. In reasoning about temporal relations we often use relations between clauses containing stative information given at different points in the text. We exploit the logical differences between activities that may end and states described by perfect tenses which, once they have started, continue forever. Any satisfactory account of temporal reasoning should do justice to the fact that certain ways of giving stative information indicate that it persists through changes in context. Of course, knowledge of the world may in addition constrain the temporal relations, as causal structure is directly related to temporal relations. But in this paper we attempt to maintain a clear distinction between knowledge of language and knowledge of the world. Semantic theory should only account for the first, assuming lexical meaning is fixed. In this paper we use the framework of Dynamic Aspect Trees (Seligman & ter Meulen, 1993,
and ter Meulen, 1995) to characterize when situated conclusions with different verbal inflections are valid in dynamic interpretation.

2. Dynamic Aspect Trees and Chronoscopes

This section introduces the dynamic representation of temporal anaphora in Dynamic Aspect Trees. Examples are discussed at first informally, to illustrate their workings and connect the English clauses to their representation in DATs. Further formalization of their syntax and semantics characterize the system more precisely, so that the notion of situated entailment can be properly defined.

2.1 Examples

(1) A car hit the fence. The driver was killed. The police arrived.

The DAT resulting from the interpretation of the first sentence of (1) is displayed in Fig. 1.

![Figure 1. First partial DAT for (1)](image)

This DAT is merged with the DAT for the second sentence, taking the hit-node as the current node and expanding the given DAT with a new plug (closed node) to represent the event of the driver being killed.

![Figure 2. Second partial DAT for (1)](image)

Fig. 3 contains the final DAT for (1), incorporating the third node representing the event of the police arriving.

![Figure 3. Complete DAT for (1.1)](image)

Perfect tense clauses are descriptions of perfect states and hence do not introduce new nodes that make the DAT grow. The resulting DAT is given in Fig. 4.

![Figure 4. DAT for (1')] (image)

From Fig. 4, but not from Fig. 5, we infer

(2) The driver was killed before his car hit the fence and before the police arrived.

If what happened is correctly represented in Fig. 5, the killing of the driver could well have taken place after the car hit the fence, but before the police arrived. That course of events is not a possible interpretation for (1'), as the perfect tense requires the event that caused the perfect state to arise to have ended before the state was described. I.e. killing the driver must have ended before the car hit the fence, leaving wide open the question what killed him.

2.2 Chronoscopes

In shifting the given chronoscope to introduce a new sister-node to the given current node, the interpretation backs up to the lowest node compatible with the new information, ensuring also that the new information is compatible with any higher node on the path to the root of the DAT. This process is comparable to the minimal revision of some dynamic systems, but it uses the tree structure to determine the order in which descriptive information is subtracted. The paths connecting a given node to its root in a DAT are called chronoscopes, that prove to be very useful instruments in temporal reasoning. A chronoscope is called the current chronoscope just in case it contains the unique current node. Compatibility of information is partly lexically determined, partly by the presuppositions and entailments between labels within the same chronoscope. Obviously a label with a positive polarity is incompatible with the label constituted of the same relation and arguments, but containing a negative polarity. As a relation between labels, compatibility is obviously reflexive, symmetric and transitive. Furthermore, if T is compatible with T' and with T'' independently, and T' is compatible with T'', then T must be compatible with T'&T''. Retracting a chronoscope in a DAT to create a new one is a simple recursive process. It terminates at the lowest ancestor node labelled with compatible information, while...
plugging its child carrying information that was still incompatible. The creation of a new chronoscope requires a certain amount of reasoning, checking of compatibility of descriptive information and various inferential processes.

2.3 Construction Rules

Based on these examples of how DATs get constructed by the interpretation of a text, the following rules may now be formulated, assuming that all information is classified in one of the four aspectual classes.

**Rules for DAT-representation**

1. If the new descriptive information is a state-type, append the label as a sticker to the current node, if it is a plug, but append the label to the next node, if the current node is a hole.
2. If the new descriptive information is a transition from start to end, introduce a new hole, make it the current node, append the label to it.
3. If the new descriptive information is a transition from start to finish, introduce a new plug, make it the current node, append the label to it.
4. If the extension of the DAT resulting from applying rules 1-4 is inconsistent, then plug the lowest dominating node labelled with an inconsistent label, make it the current node and reapply the rules.

The nature of the current node determines, as we have seen, where the new node is located. For future reference, these rules are stated here as well.

**Rules for introducing new nodes**

(i) if the current node is a hole, represent the new node as child of it.
(ii) if the current node is a plug, represent the new node as sister to it.

Plugging of an internally structured node hides its internal structure, rendering it inaccessible in the new current chronoscope. The effect of this plugging up is that all the information labelling the inaccessible nodes must be reported as static information using the perfect tense, except for the stickers that may be transmitted to a node in the new current chronoscope. Plugs, terminating a chronoscope, are self-dual, they have no internal structure in the given perspective.

3. Reasoning with DATs

In reasoning with a DAT we rely on a fundamental property of the situations that support them, called persistence. If a situation $s_2$ supports DAT$_1$, for instance, and DAT$_2$ grows out of DAT$_1$ by application of the rules formulated above, and $s_2$ supports DAT$_2$, then $s_2$ supports DAT$_1$. Growing DATs models the accumulation of information in processing natural language input. Persistence means that the information represented in an earlier stage of construction of a DAT is not lost, when the interpretation is continued, although it may not remain accessible from any later current node. The way the available information may be expressed in conclusions depends on where the current node is in the DAT and its relation to the node labelled with the information the conclusion is based on. The given DAT may be affected by incorporating new information, not merely by growing new nodes, but also by plugging up holes or unplugging plugs. The interpretation process defines a non-monotonic relation between DATs.

The list of inferences based on (1) is given below in (3). (3) inferences from (1.1) (* indicates invalid inference)

a. The car hit the fence before the driver was killed
b. *The car hit the fence when/after the driver was killed
c. The driver was killed when/after the car had hit the fence
d. The car had hit the fence before/when the driver was killed
e. *The driver had been killed after/while the car hit the fence
f. The police arrived after the driver was killed
g. The police arrived when the driver was dead
h. The police arrived when the car had hit the fence
i. The car having hit the fence, the police arrived
j. The car having hit the fence, the police arrived
k. The police arrived when the driver was dead
l. The police arrived after the driver was killed
m. The police arrived after/while the car hit the fence
n. The police arrived after/while the car hit the fence

The completed DAT for (1) was given in Fig. 3. Its current node is the one, carrying the information that the police arrived, immediate left sister of the source. The conclusions also describe the current node, if the argument is valid. In this sense, reasoning with the information represented in a DAT is always dependent upon the current node, and this makes the temporal reasoning a form of situated reasoning. The current node is the point of departure, as it were, in the process of attempting to verify the conclusion. For (3a) to be true, given this DAT for (1), there must be a node in the current chronoscope dominating the two nodes labelled with the information expressed by simple past clauses, linked by the temporal adverbial before. This adverbial constrains the relation between these nodes with this common ancestor: the node representing the car hitting the fence should be a left descendant of the common ancestor relative to the node representing the driver getting killed. This is obvious in Fig. 3: the common ancestor is the root, its left descendant is indeed the node representing the car hitting the fence relative to the node representing the killing of the driver. If the clause-linking adverbial had been when or after, as in (3b), the conclusion would not follow from the DAT in Fig.3, as they would require the hitting node respectively to be in the same chronoscope as the killing node or be a right descendant of its common ancestor. In (3c-e) the perfect clause describes the enduring state that began after the hitting ended. That state is represented by a sticker appended to the current node, when we interpret the conclusion. The adverb when requires that the related nodes be in the same chronoscope. Using after as temporal connector, the perfect state is required to have started before action reported in the simple past tense clause started. Using before/when as in (3d) is similar, when preceded with descriptions in perfect tense. But, as
evident in (3 e), to the label of the current node you cannot append a sticker describing the state resulting from a later event.

Verifying a conclusion adds the label \( \text{PERF} (T) \) to the current node and searches for a non-empty intersection of the current chronoscope with a chronoscope containing a left descendant node labelled \( T \). The DAT representation suggests a rather simple systematic terminating search-algorithm: back up to the first node dominating the current node and see whether it is contained in any chronoscope containing the desired node; if so, the conclusion is verified, if not, back up to the next higher node in the current chronoscope, repeat until the entire DAT is searched. If you have not been able to verify the conclusion, the argument is invalid. In verifying (3 f-i), the current node is labelled with the information that the police arrived. All that needs verifying is that it is a right descendant of a common ancestor of the node representing the killing (3 f). The causal consequences of that node, i.e. the state of the driver being dead, are stickers portable within the current chronoscope (3 g), and the current chronoscope intersects with the chronoscope containing the node that the car hit the fence (3 h). For the gerundive perfect conclusion in (3 i) we need to verify that the perfect type labels a node in the current chronoscope. For \( \text{PROG} (T) \) conclusions, it is required that an ancestor of the current node, i.e. in the current chronoscope, is labelled \( T \) or \( \text{PROG} (T) \) again.

These examples illustrated how inferences are made using a DAT representation of the information given in the premises. Reasoning with a DAT consists of systematic operations or procedures to verify conclusions from its current node. This operationalization of inferential processes clarifies what is gained by such graphic representations of information over pure symbolic logical form representations. The form of a DAT and the location of its current node steers the verification of the conclusion by providing a systematic search procedure, guaranteed to terminate either in a verification or in a rejection of the conclusion. Drawing a conclusion never adds a new node to the given DAT for its premises, but the label coding the conclusion is appended to the current node from when the verification succeeds. Current research formalizes the domain-specific constraints used in temporal reasoning as structural modalities in a dynamic temporal logic, using Gentzen style sequents to express the inference rules.

4. Formalization

Types that label the nodes of a DAT are formed by the following rules, recursively specifying the class of types \( \text{TYPE} \).

**Definition of \( \text{TYPE} \)**

(i) \( T \) is a basic type in \( \text{TYPE} \) iff. \( T \) is a sequence of an \( n \)-ary relation \( R \), \( n \) objects \( a_1, ..., a_n \), and a positive or negative polarity \( + \) or \( - \).

(ii) \( T \) is a parametric or parametrized type in \( \text{TYPE} \) iff. \( T \) is a basic type in which a relation or an object is replaced by a relation parameter \( R \) or an object parameter \( a \), respectively.

(iii) if \( T \) is in \( \text{TYPE} \) and \( x \) is an object parameter then \( x_T \) is a restricted object parameter. If \( T \) contains \( x_T \), \( T \) is a restricted parametrized type in \( \text{TYPE} \).

(iv) if \( x \) is a parameter and \( T \) is in \( \text{TYPE} \), then \( [xT] \) is a role. If \( T \) is a restricted parametrized type, all parameters in the restriction must occur in the role-type to the left of \( t \).

(v) if \( T \) and \( T' \) are in \( \text{TYPE} \), then so is the conjoined type \(< T \& T' >\) and the conditional type \(< T \Rightarrow T' >\).

(vi) closure.

Any two types are compatible iff. there is a situation that supports both within the same chronoscope. Obviously, any type is compatible with itself, and incompatible with its negative counterpart with a negative polarity or any of its entailments. Two sets of types are compatible iff. their union is compatible; and a single type \( T \) is compatible with a set \( S \) of types just in case \( S \cup \{ T \} \) is compatible.

A DAT consists of:

(i) a finite set of nodes, \( N = \{ n, n', ..., n_m \} \)

(ii) a function \( \delta_N \) from \( N \) to \( N^* \), where \( N^* \) is the set of non-repeating finite sequences of \( N \), assigning to each node \( n \) a sequence of nodes, its children or immediate dependents \( \delta_N (n) \).

(iii) a function \( \pi_N \) from \( N \) to \( N \), assigning an arrow pointing to a node \( n \) from its immediately dominating node, its parent \( \pi_N (n) \).

(iv) a subset \( H_N \) of \( N \), the Holes; \( N - H_N \) is the set \( P_N \) of plugs.

(v) a function \( \alpha_N \) from \( N \) to the powerset of \( \text{TYPE} \), assigning to each node a set of types.

(vi) a distinguished node \( c_N \) in \( N \), the current node, and another distinguished node \( s_N \) in \( P_N \), the source node.

such that

1. \( \forall n, n' \) in \( N \), \( n \) is in \( \delta_N (n') \) iff. \( \pi_N (n') = n' \)
2. There is one and only one node, the root, that is its own ancestor \( n \) is an ancestor of \( n' \) iff. \( \exists n_1, ..., n_m (m \geq 2) \) such that \( n_i = \pi_N (n_i + 1) \), \( n = n_1 \) and \( n' = n_m \).
3. The source node is the terminal plug of the right most chronoscope.
4. The set of all types labelling the ancestors of a node \( n \) (i.e. \( \cup \{ \alpha_N (n) \} \) \( n \) is an ancestor of \( n \) ) is compatible with those types labelling \( n \) itself.

Information accumulation is simulated by a growing DAT. It is defined as an ordering on DATs as follows: \( D_1 < D_2 = D_1 \cup \{ cD_2 \} \) and either

1. (Hole rule) \( cD_1 \) is in \( HD_1 \) and each of \( \delta_D \), \( \pi_D \), \( HD_2 \) and \( cD_2 \) extends the corresponding function of \( D_1 \) with the single exception that \( \delta_D (cD_1) = < cD_2 > \), or
2. (Plug rule) $cD_1$ is in $PD_1$ and each of $\delta D_2 , \pi D_2 , HD_2$ and $\alpha D_2$ extends the corresponding function of $D_1$ with the single exception that $\delta D_2 (\pi D_1 (cD_1))$ is obtained by appending $cD_2$ to the end of $\delta D_1 (\pi D_1 (cD_1))$, or

3. (Filler rule) $\alpha D_2 (cD_2)$ is incompatible with the types assigned to the ancestors of $cD_1$; and $\exists D' I < D_2$ with the same nodes as $D_1$ and the same functions $\delta , \pi$ and $\alpha$ but with $cD' I$ as ancestor of $cD_1$ and $cD' I$ is not in $HD' I$.

4. (Sticker rule) $cD_1$ is in $ND_1$ and each of $\delta D_2 , \pi D_2 , ND_2$ is identical to its corresponding function of $D_1$, but $\alpha D_2$ extends the corresponding function of $D_1$ and $cD_1 = cD_2$, if $cD_1$ is in $PD_1$, and otherwise $\alpha D_2$ is extended into $\alpha D_3$ after application of the hole or filler rule.

Define $\prec$ to be the transitive closure of $<$, i.e. $D \prec D'$ iff. there are $D_1 , ..., D_n$ such that $D_i < D_{i+1}$ and $D = D_1$ and $D' = D_n$.

Let $\delta$ be the DAT with a single node $\delta$, $\pi D (\delta) = \delta$, $\pi D (\delta) = \delta$, $\alpha D (\delta) = \delta$, $\alpha D (\delta) = \delta$. The class of wellformed DATs consists of only those DATs $D$ such that $\delta \prec D$.

DATs are interpreted in event-structures, consisting of events with their natural temporal inclusion and precedence ordering and constrained by some additional temporal conditions. A $DAT$-frame consists of a set of events $E$ ordered by temporal inclusion $\to (x \to y$ - $y$ is a temporal part of $x$ and temporal precedence $< (x < y$ - $x$ occurs before $y)$, with an assignment to each $T$ in $TYPE$, of a set of events $[[ T ]]$, the extension of $T$, such that the temporal inclusion is a p.o., precedence is a strict p. o. and constrained by:

- monotonicity: if $y \to x$ and $y < z$ then $x < z$
- convexity: if $x < y < z$ and $u \to x$ and $u \to z$
  then $u \to y$

DATs are interpreted in such event frames by embeddings, mapping nodes to events preserving the temporal relations and satisfying certain additional conditions.

**Definition of embedding**

A function $f$ mapping a DAT into a model $M$ is an embedding when:

(i) for every arrow $\pi N (n) \to n$, $f (\pi N (n)) \to f(n)$
(ii) if $n$ commands $n'$, then $f(n) < f(n')$
(iii) $f(n) = T$, where $T$ is the type labelling $n$.

The event-structures $S$ that are suitable for temporal reasoning must in addition satisfy certain domain specific constraints.

**Definition of situated entailment**

Let $D$ be a DAT for the premises $T_1, ..., T_n$ and let $c$ be its current node,

$T_1, ..., T_n \vdash T$ iff.

for all $S$ and all embeddings $f$ of $D$ into $S$, if $T_1, ..., T_n$ describes $f(D)$, then $f(c)$ is of type $T$.

**References**