Situations as strings

Tim Fernando
Trinity College Dublin
WoLLIC, Stanford
July 2006

**Starting point**  A proposition is a set of worlds

**Idea**  Sharpen *world* to *situation* (Barwise, …)
Situations big and small

<table>
<thead>
<tr>
<th>natural language semantics</th>
<th>world $w$</th>
<th>event $e$</th>
</tr>
</thead>
<tbody>
<tr>
<td>L. Schubert on situations</td>
<td>modality</td>
<td>temporality</td>
</tr>
<tr>
<td>logical semantics</td>
<td>Carnap, Montague</td>
<td>Davidson, ...</td>
</tr>
<tr>
<td>topology</td>
<td>truth-conditional</td>
<td>proof-theoretic</td>
</tr>
<tr>
<td></td>
<td>set-theoretic models</td>
<td>constructive types</td>
</tr>
<tr>
<td></td>
<td>point-set</td>
<td>point-less</td>
</tr>
</tbody>
</table>

$w \models \text{may-have-rained-yesterday}$ iff $(\exists w' Rw)\ w' \models \text{rained-yesterday}$

$w \models \text{rained-yesterday}$ iff $(\exists e \sqsubseteq w)\ e : \text{rained-yesterday}$
Below *Propositions-as-types* (PaT)

PaT says nothing special about
- what proofs of atomic formulas are
- time or change (inertia/frame problem: McCarthy and Hayes)

**Subatomic semantics** (T. Parsons)

*the study of those “formulas in English” that are treated as atomic formulas in most logical investigations of English. The main hypothesis to be investigated is that simple sentences of English contain subatomic quantification over events.*

Tense and aspect

(1) Pat had been gaining weight.
(2) Pat read the newspaper for/in an hour.
Case Study: Linear Temporal Logic (LTL)

Kripke frame \((\mathbb{Z}, <)\) with present 0 and temporal precedence <

\[ i \in \text{past} \quad \text{iff} \quad i < 0 \]

\[ i \in \text{future} \quad \text{iff} \quad 0 < i \]

Valuation \(x : \mathbb{Z} \rightarrow 2^P\) given a set \(P\) of atomic propositions

\[ x \models p \quad \text{iff} \quad p \in x(0) \]

\[ x \models \text{next}(\varphi) \quad \text{iff} \quad x^1 \models \varphi \]

where \(x^i = (\lambda n \in \mathbb{Z}) \ x(i + n)\)

\[ x \models \varphi \text{ since } \psi \quad \text{iff} \quad (\exists i < 0) \ x^i \models \psi \text{ and } x^j \models \varphi \text{ for } i < j \leq 0 \]
From valuations (points) to strings (basic open sets)

\[ x \models p \land \text{next}(q) \quad \text{iff} \quad p \in x(0) \text{ and } q \in x(1) \]

\[ = \quad \text{iff} \quad \text{now, } p \models q \subseteq x \]

\[ x \models p \text{ since } q \quad \text{iff} \quad (\exists s \subseteq x) \ s \in \underbrace{q \models p^*}_{\text{now, } p} \]

L(p \text{ since } q)

Analyze \( \varphi \) as a language \( L(\varphi) \subseteq (2^P \cup \{\text{now}\})^* \) so that

\[ x \models \varphi \quad \text{iff} \quad (\exists s \subseteq x) \ s \in L(\varphi) \]

Conflating strings with languages,

\[ L(p \land \text{previous}(q)) = q \models \text{now, } p \]

\[ L(p \text{ until } q) = \text{now, } p \models p^* \models q \]
Infinite strings via lazy evaluation

\[ x \models \text{always}_>(\varphi) \quad \text{iff} \quad (\forall i \geq 0) \ x_i \models \varphi \]

\[ L(\text{always}_>(p)) \approx \text{now}, p \ p \ p \ p \ldots \]

\[ = \lim_{i \to \infty} \text{now}, p \ p^i \]

Finite approximations \( s \in (2^\Phi)^* \) where \( \Phi \supseteq P \cup \{\text{now}\} \)

\[
\begin{align*}
\text{now,always}_>(p) & \leadsto \text{now}, p \ \text{always}_>(p) \\
& \leadsto \text{now}, p \ p \ \text{always}_>(p) \\
& \quad \vdots \\
& \leadsto \text{now}, p \ p^i \ \text{always}_>(p) \quad \leadsto \ldots
\end{align*}
\]
Finite-state issues

\[ \text{fut}(\varphi) \leadsto \square^+ \varphi \]

\[ \varphi \text{ until } \psi \leadsto \varphi^+ \psi \]

\[ \varphi \land \psi \leadsto \varphi, \psi \]

\[ (p \land \text{fut}(q)) \text{ until } r \leadsto p, \text{fut}(q)^+ r \]

\[ \cdots \cap p^+ r q^+ = \sum_{i \geq 1} \sum_{1 \leq j \leq i} p^i r q^j \]

non-regular!

Regular sublanguage \( p^+ r q \) and for \( p, \text{fut}(q)^+ r \),

\[ p^+ (q, r + r \square^* q) \]
Constraints and subsumption

\[
\begin{align*}
\text{fut}(\varphi) & \implies \Box \varphi + \Box \text{fut}(\varphi) \\
\varphi \text{ until } \psi & \implies \varphi \psi + \Box \varphi \text{ until } \psi \\
\text{always} > (p) & \implies p \text{ always} > (p)
\end{align*}
\]

\[s \in A \implies B \quad \text{iff} \quad \text{any stretch of } s \text{ that contains a string in } A \text{ contains one in } B\]

Containment as subsumption \( \supset \)

\[
a_1 \cdots a_n \supset b_1 \cdots b_m \quad \text{iff} \quad n = m \text{ and } a_i \supseteq b_i \text{ for } 1 \leq i \leq n
\]

\[
p, q \supset p \supset p + q
\]

\[
L \supset L' \quad \text{iff} \quad (\forall s \in L)(\exists s' \in L') \ s \supseteq s'
\]
Regularity of constraints and conciseness

No \((A \land \neg B)\)-counter-examples

\[
A \Rightarrow B = (2^\Phi)^*(A^\triangleright \cap \overline{B^\triangleright})(2^\Phi)^*
\]

where

\[
L^\triangleright = \{ s \mid (\exists s' \in L) \ s \triangleright s' \} .
\]

Apply the constraint \(A \Rightarrow B\) to \(L\)

\[
(A \Rightarrow B) \cap L^\triangleright
\]

and take \(\triangleright\)-minimal strings.

\[
L_{\triangleright} = \{ s \in L \mid (\forall s' \in L) \ s \triangleright s' \text{ implies } s = s' \} \\
= L - \{ s \mid (\exists s' \in L - \{s\}) \ s \triangleright s' \}
\]

**Fact.** \(L^\triangleright\) and \(L_{\triangleright}\) are regular if \(L\) is.
Inertia and force (always finitarily)

ϕ is inertial if it persists unless forced not to

\[ \quad \square \, \varphi \quad \Rightarrow \quad \square \varphi + \, \square \, \neg \varphi \]

\[ \quad \square \varphi \quad \Rightarrow \quad \varphi \square + \, \square \, \varphi \]

\[ f\neg \varphi \approx \text{there is a force against } \varphi \]

\[ f\varphi \approx \text{there is a force for } \varphi \]

(3) Pat stopped the car before it hit the tree.

\[ \cdots \text{still(car)} \quad \text{still(car)} \quad \cdots \]

(4) Pat left Dublin but is back.

\[ \text{?Pat has left Dublin but is back.} \]
Prior 1967

the usefulness of systems of this sort does not depend on any serious metaphysical assumption that time is discrete; they are applicable in limited fields of discourse in which we are concerned with what happens in a sequence of discrete states, e.g. in the workings of a digital computer.

Discreteness

- in computation
- in planning
- from finiteness

To show: we can take time to be the real line
Events as strings, reduced

\[ L(\text{rain from dawn to dusk}) = \text{rain, dawn}\text{rain} + \text{rain, dusk} \]

Is each string \( \text{rain, dawn}\text{rain}^i\text{rain, dusk} \) a distinct event?

For \( i \geq 1 \), reduce to \( \text{rain, dawn}\text{rain}\text{rain, dusk} \)

\[ \text{interval reduction} \ ir(s) \ \text{of} \ s \]

\[
ir(s) = \begin{cases} 
    s & \text{if} \ \text{length}(s) \leq 1 \\
    ir(as') & \text{if} \ s = aas' \\
    air(a's') & \text{if} \ s = aa's' \ \text{where} \ a \neq a' 
\end{cases}
\]

\[
ir(□□□p□p□p□□□q□□) = □□p□q□
\]
Events as inverse limits

Turn any finite sequence of real numbers

\[ r_1 < r_2 < \cdots < r_n \]

into the string

\[ \square r_1 \square r_2 \square \cdots \square r_n \square \]

to approximate the real line \((\mathbb{R}, <)\) by finite subsets \(X\) of \(\mathbb{R}\).

\[
\begin{align*}
\bigcap_X (a_1 \cdots a_n) &= (a_1 \cap X) \cdots (a_n \cap X) \\
\bigcap\{r_2, r_4\}\big(\square r_1 \square r_2 \square r_3 \square r_4 \square\big) &= \square r_2 \square r_4 \square \\
\text{ir}_X(s) &= \text{ir}(\bigcap_X(s)) \\
\text{ir}_{\{r_2, r_4\}}\big(\square r_1 \square r_2 \square r_3 \square r_4 \square\big) &= \square r_2 \square r_4 \square \\
\lim(2^X)^* &= \{ (s_X)_{X \in \text{Fin}(\mathbb{R})} \in \prod_{X \in \text{Fin}(\mathbb{R})} (2^X)^* \mid \} \\
s_{X'} &= \text{ir}_{X'}(s_X) \text{ for } X' \subseteq X \in \text{Fin}(\mathbb{R}) \}
\]
Dynamic semantics (DRT, DPL): not strictly computational
(negation = complement of the halting problem)

<table>
<thead>
<tr>
<th>logical semantics</th>
<th>truth-conditional</th>
<th>proof-theoretic</th>
</tr>
</thead>
<tbody>
<tr>
<td>topology</td>
<td>point-set</td>
<td>point-less (⊑)</td>
</tr>
<tr>
<td>formal verification</td>
<td>model-checking</td>
<td>theorem-proving</td>
</tr>
</tbody>
</table>

In nl semantics, access to suitable model/point is problematic.

No compiler for English.

Partiality is crucial — there’s nothing partial about a point.

Regard proofs as hypothetical (from a context of variable typings).