Journal of Semantics 25: 321–344
doi:10.1093/jos/ffn002
Advance Access publication June 3, 2008

Branching from Inertia Worlds

TIM FERNANDO
Trinity College Dublin

Abstract

The notion of inertia is explicated in terms of forces recorded in snapshots that are strung together to represent events. The role inertia worlds were conceived to serve in the semantics of the progressive is assumed by a branching construct that specifies what *may* follow, apart from what follows.

1 INTRODUCTION

Sentences such as (1) illustrate the observation in Dowty (1979) that an event in progress at time I in world w may fail to culminate in w.

(1) Pat was losing when the match took an unexpected turn.

Dowty employed the notion of an (I, w)-*inertia world* to insure that an event e in progress at I in w culminates somewhere—namely, in every (I, w)-inertia world. We may expect e to culminate in w inasmuch as we may expect w to be an (I, w)-inertia world. As there is no law making w an (I, w)-inertia world, however, e need not culminate at w. But then can we assume (I, w)-inertia worlds exist? Surely we may assume that there are events other than e in progress at I in w. Can we be certain *not* one of them clashes with e down the line? Landman (1992) credits the following example to Roger Schwarzschild.

> Suppose I was on a plane to Boston which got hijacked and landed in Bismarck, North Dakota. What was going on before the plane was hijacked? One thing I can say is: 'I was flying to Boston when the plane was hijacked.' This is reasonable. But another thing I could say is: 'I was flying to Boston. Well, in fact, I wasn't, I was flying to Bismarck, North Dakota, but I didn't know that at the time.' And this is also reasonable. (pp. 30–1)

With this (or perhaps a different example) in mind, let us suppose another event e' were to be in progress at I in w that culminates only in a world where e does not. Then, the requirement that e and e' culminate at *every* (I, w)-inertia world would mean there is *no* (I, w)-inertia world (rendering talk of inertia worlds pointless). This suggests refining the relativization (I, w) on inertia worlds to discriminate between events e and e' that are headed for a conflict. The alternative is to deny that such events can be in progress at the same pair (I, w). Stepping back, we might ask what (on earth) an inertia world is. Are our intuitions about inertia worlds coherent and reliable enough that we can be comfortable with a semantic account that treats inertia worlds as primitive?

Rather than take inertia worlds for granted, the present work pursues a constructive approach, building worlds bottom-up from temporal propositions. I shall refer to these propositions as *fluents*, following the custom in artificial intelligence since McCarthy & Hayes (1969) and more recently in linguistic semantics (van Lambalgen & Hamm 2004). Very briefly,

(i) the notion of inertia is fleshed out against fluents (as opposed to worlds), some of which are assumed to be inertial while others describe forces, and
(ii) a world is formed from the events that happen in it

where an event is formulated as a string of sets of fluents (Fernando 2004). The analysis of inertia and worlds in (i)–(ii) stops short of addressing modal matters such as the implication (2b) of a natural reading of (2a).

(2) a. Pat stopped the car before it hit the tree.
  b. The car did not hit the tree but it might well have.

To interpret (2b), we need structures recording what *may* happen, over and above what *does* happen. Accordingly, strings are extended to branching strings, which are closed under not only concatenation but also a second binary operation. That second operation yields branches recording some of the ways things *may* turn out under historical necessity (e.g. Thomason 1984). Branching is, I claim, implicit in the step from a world w to the set of (I, w)-inertia worlds. What's more, it is branching, and not some notion of inertia world, that figures in the semantics of the progressive formalized below.

To motivate that formalization, section 2 examines entailments $\vdash$ of progressive forms that are associated with the appropriateness of temporal modification by *for*, as opposed to *in* (Vendler 1967).

(3) a. Pat was walking $\vdash$ Pat walked
  b. Pat walked for (?in) five minutes.

(4) a. Pat was walking home $\nvdash$ Pat walked home
  b. Pat walked home in (?for) five minutes.

As for inertia, section 3 investigates its role in the contrast between the simple past and the present perfect (e.g. Steedman 2000).

(5) a. Pat left Dublin but is back (in Dublin).
b. ?Pat has left Dublin but is back (in Dublin).

Inertial laws are laid down specifying the persistence of inertial fluents in the absence of forces. We work with events as in Parsons (1990) and Landman (1992), and follow the latter in looking at what might become of an event in progress, for a glimpse at modal implications of the progressive related to (2). That is, we explore 'continuations branches' as in Landman's semantics of the progressive, but with*out* appealing to 'a Stalnaker-style theory of subjunctives' or (for that matter) worlds. Instead, continuation branches are construed perspectivally on the basis of the structure on events induced by their string representations. Entailments can be read directly off strings, making it unnecessary to resort to worlds to define entailments in terms of truth at worlds. Rather than sets of worlds as propositions, we study sets of strings as event types. And rather than defining entailment $\vdash$ between propositions p and q by the subset relation $\subseteq$

$$p \vdash q \overset{\text{def}}{\Leftrightarrow} p \subseteq q \text{ (as sets of worlds)},$$

we refine $\subseteq$ to a subsumption relation $\trianglerighteq$ (defined in the next section) between sets of strings. The refinement needed here is the key to the analysis of branching in section 4.

2 EVENT TYPES AS SETS OF STRINGS

Underlying our representation of events as strings is the intuition that an event is 'a series of snapshots' (Tenny 1987). Equating a snapshot with a set of fluents that describe it, we represent an event as a string $a_1 a_2 \ldots a_n$ of sets a_i of fluents describing n successive moments. We finesse questions about the choice of successive moments and string length n by working with a *set* of strings. For instance, we might associate *rain from dawn to dusk* with the set

$$[\text{rain, dawn}][\text{rain}]{*}[\text{rain, dusk}] = \{[\text{rain, dawn}][\text{rain}]^n[\text{rain, dusk}] \mid n \geqslant 0\}$$

of strings [rain, dawn][rain]^n[rain, dusk] with $(n + 2)$ snapshots, all of rain, the first at dawn and the last at dusk. The different values of n correspond to different levels of temporal granularity (the larger the n, the finer the grain).[1] We use square brackets [and], instead of curly

[1] Whether or not these infinitely many strings should be thought of as distinct events depends on how we interpret the temporal relation between successive snapshots. We will sidestep this question by focusing less on the conception of an event-as-string and more on that of an event-type-as-string-set.

braces, to enclose a set of fluents when it is intended as a snapshot. Boxes are arguably preferable to square brackets, but are a nuisance to typeset. Hence, we will refrain from drawing boxes except for the empty snapshot, which we write $\Box$ instead of []. Beyond reinforcing the filmstrip metaphor, this helps distinguish the string $\Box$ of length 1 (containing no fluents) from the empty string set $\emptyset$ (containing no strings). We refer to a string set as a language (as in formal language theory), and write * for Kleene star, L^+ for LL^* and $+$ for non-deterministic choice. We collect the fluents in a set Φ, turning snapshots into elements of the powerset Pow(Φ) that serves as our alphabet. Snapshots here are (unlike McCarthy & Hayes 1969) partial: $\Box$ is as much a part of $[\varphi]$ as it is of $[\bar{\varphi}]$, where $\bar{\varphi}$ is the negation of φ. Henceforth, we assume that every fluent φ comes with another fluent $\bar{\varphi} \neq \varphi$ and that $\bar{\bar{\varphi}} = \varphi$. A snapshot may be φ-underdefined in that it contains neither φ nor $\bar{\varphi}$ (e.g. $\Box$), or else it may be φ-overdefined, containing both φ and $\bar{\varphi}$. Whereas φ-underdefinedness simply reflects the partiality of observations, φ-overdefinedness indicates that the observation has gone awry. This suggests restricting the alphabet Pow(Φ) to the family

$$nc(\Phi) \overset{\text{def}}{=} \{a \in \text{Pow}(\Phi) | (\forall \varphi \in a) \bar{\varphi} \notin a\}$$

of *non-contradictory* snapshots (assembled from Φ). Nevertheless, we will define operations and relations on languages over the alphabet Pow(Φ), as it is easy enough to intersect a language $L \subseteq \text{Pow}(\Phi)^*$ with $nc(\Phi)^*$ (and indeed other constraints) to weed out spurious descriptions.

A natural conjunction of languages L, L' over Pow(Φ) is the *superposition* L & L' *of* L *and* L' obtained from the componentwise union of strings in L and L' of the same length

$$L \,\&\, L' \overset{\text{def}}{=} \bigcup_{n \geqslant 0} \{(a_1 \cup b_1) \ldots (a_n \cup b_n) | a_1 \ldots a_n \in L \text{ and } b_1 \ldots b_n \in L'\}$$

(Fernando 2004). For example, we have (6).

(6) $[\text{rain, dawn}][\text{rain}]^*[\text{rain, dusk}] = [\text{rain}]^+ \& [\text{dawn}]\Box^*[\text{dusk}]$
$= [\text{rain}]^+ \& ([\text{dawn}]\Box^+ \& \Box^+[\text{dusk}])$

To capture the growth of information from &, let us say that L *subsumes* L' and write $L \trianglerighteq L'$ if the superposition of L and L' includes L

$$L \trianglerighteq L' \quad \overset{\text{def}}{\Leftrightarrow} \quad L \subseteq L \& L'$$

(roughly: L is at least as informative as L'). Conflating a string s with the singleton language $\{s\}$, we get (7) and (8). (7) states that $\unrhd$ holds between strings of the same length related componentwise by inclusion. (8) says L subsumes L' exactly if each string in L subsumes some string in L'.

(7) $a_1 \ldots a_n \unrhd b_1 \ldots b_m$ iff $n = m$ and $a_i \supseteq b_i$ for $1 \leqslant i \leqslant n$

(8) $L \unrhd L'$ iff $(\forall s \in L)(\exists s' \in L')s \unrhd s'$

As (9) illustrates, we may regard a comma inside a box as conjunction, and plus between strings as disjunction.

(9) $[\varphi, \psi] \unrhd [\varphi] \unrhd [\varphi] + [\psi]$

That is, as a type with instances $s \in L$, a language L is essentially a disjunction $\bigvee_{s \in L} s$ of conjunctions s (Fernando 2004).

Subsumption $\unrhd$ and superposition & go hand in hand in specifying languages $\mathcal{L}(A)$ for English sentences A such as those in (3) and (4), repeated here.

(3) a. Pat was walking $\vdash$ Pat walked
b. Pat walked for (?in) five minutes.

(4) a. Pat was walking home $\nvdash$ Pat walked home
b. Pat walked home in (?for) five minutes.

The general idea is to reduce entailment to subsumption and trace the unacceptability of A to the disjointness of $\mathcal{L}(A)$ from $nc(\Phi)^*$. (Note that a language L is disjoint from $nc(\Phi)^*$ iff every string $a_1 \ldots a_n$ in L is contradictory in that for some $\varphi \in \Phi$, both φ and $\bar{\varphi}$ belong to some a_i.)

(10) a. $A \vdash B$ because $\mathcal{L}(A) \unrhd \mathcal{L}(B)$
b. $?A$ because $\mathcal{L}(A) \cap nc(\Phi)^* = \emptyset$

Leaving inflection from tense and aspect out for the moment, we assume a fluent home(p) (saying Pat is home) is negated in the language $\mathcal{L}$(Pat walk home) until the very end.

(11) $\mathcal{L}(\text{Pat walk home}) \unrhd \overline{[\text{home}(p)]}^{+}[\text{home}(p)]$

(11) says $\mathcal{L}$(Pat walk home) is home(p)-telic, where we call a language L *φ-telic* if $L \unrhd [\bar{\varphi}]^*[\varphi]$.[2] More generally, we collect fluents that

[2] This analysis makes sense only if we assume some bounded temporal granularity, with discernibly different increments. A discrete as opposed to continuous conceptualization of change saves us from pinning the precise moment of change (or measuring the duration of a state/event) to an arbitrary (arguably meaningless) degree of precision.

appear at the end of every L-string in the set ω_L of fluents φ such that $L \unrhd \Box^*[\varphi]$

$$\omega_L \overset{\text{def}}{=} [\varphi \ | L \unrhd \Box^*[\varphi]]$$

and define L to be *telic* if $L \unrhd \overline{\omega_L}^* \Box$, where the *negation* $\bar{a} \subseteq \text{Pow}(\Phi)$ *of* $a \subseteq \Phi$ is (in accordance with De Morgan) the disjunction (given by +) of negations of fluents in a

$$\overline{[\varphi_1, \ldots, \varphi_n]} \overset{\text{def}}{=} [\overline{\varphi_1}] + \cdots + [\overline{\varphi_n}]$$

(with $\overline{\Box} \overset{\text{def}}{=} \emptyset$). Clearly, if L is φ-telic then L is telic. As for $\mathcal{L}(\text{Pat walk})$, let us call L *iterative* if $L \unrhd \Box\omega_L^*$. Now, suppose we analyse the interval *five minutes* as in (12), with fluents $0(\tau)$ and $5\text{min}(\tau)$ saying that zero time and five minutes have (respectively) passed since time τ.

(12) $\mathcal{L}(\text{five minutes}) \unrhd [0(\tau)]\Box^+[5\text{min}(\tau)]$

Then, we can put down the contrast between (3b) and (4b) to (13), and an interpretation of temporal *in*- and *for*-modification by the superpositions in (14) with a language I representing the temporal interval.

(3) b. Pat walked for (?in) five minutes.

(4) b. Pat walked home in (?for) five minutes.

(13) a. $\mathcal{L}(\text{Pat walk home})$ is telic.
b. $\mathcal{L}(\text{Pat walk})$ is iterative.

(14) a. $in(L, I) \overset{\text{def}}{=} L \ \& \ I \ \& \ \overline{\omega_L}^* \Box$
b. $for(L, I) \overset{\text{def}}{=} L \ \& \ I \ \& \ \Box\omega_L^*$

According to (14), *in* contributes $\overline{\omega_L}^* \Box$, whereas *for* contributes $\Box\omega_L^*$. Let us say L is *durative* if $L \unrhd \Box\Box\Box^+$ (that is, every string in L has length $\geqslant 3$), and let us write $\equiv$ for the intersection of $\unrhd$ with its converse

$$L \equiv L' \overset{\text{def}}{\Leftrightarrow} L \unrhd L' \text{ and } L' \unrhd L.$$

Assuming I is durative, we derive (15) and (16), matching telic and iterative languages with temporal *in*- and *for*-modification, respectively.

(15) If L is telic then $in(L, I) \equiv L \ \& \ I$ and
$for(L, I) \cap nc(\Phi)^* = \emptyset$.

(16) If L is iterative then $for(L, I) \equiv L \ \& \ I$ and
$in(L, I) \cap nc(\Phi)^* = \emptyset$.

Next, for tense and aspect, we adopt a Reichenbachian approach that locates the event time E relative to not only a speech time S but also a *reference time* R (Reichenbach 1947). To see how R might be useful, consider the pair in (5a,b).

(5) a. Pat left Dublin but is back (in Dublin).
b. ?Pat has left Dublin but is back (in Dublin).

The position of the speech time S relative to Pat's departure from Dublin is not enough to differentiate (5a) from (5b); abbreviating $\mathcal{L}$(Pat leave Dublin) to L_0,

$$\mathcal{L}(\text{Pat left Dublin}) \trianglerighteq L_0\Box^*[S] \text{ and } \mathcal{L}(\text{Pat has left Dublin}) \trianglerighteq L_0\Box^*[S].$$

R allows us to distinguish $\mathcal{L}$(Pat left Dublin) from $\mathcal{L}$(Pat has left Dublin), coinciding with the end of L_0 in the former case, and with S in the latter.

(17) a. $\mathcal{L}(\text{Pat left Dublin}) \trianglerighteq (L_0 \,\&\, \Box^*[R])\Box^*[S]$
b. $\mathcal{L}(\text{Pat has left Dublin}) \trianglerighteq L_0\Box^*[R, S]$

Treating R and S as fluents (in Φ) while fleshing out E as a language (e.g. L_0), we derive the contrast in (5) from (17) in the next section. In general, we apply aspectual and tense operators in sequence, forming, for instance,

$$\begin{aligned}\mathcal{L}(\text{Pat left Dublin}) &= \textsc{Past}(\textsc{Simp}(L_0))\\ \mathcal{L}(\text{Pat has left Dublin}) &= \textsc{Pres}(\textsc{Perf}(L_0))\end{aligned}$$

with tense operators for the past and present applied after operators for simple and perfect aspect. If an event with temporal projection E is represented by a language L, we get three aspectual operators on L.

(18) a. $\textsc{Simp}(L) \overset{\text{def}}{=} L \,\&\, \Box^*[R]$ (i.e. R is E's right end)
b. $\textsc{Prog}_o(L) \overset{\text{def}}{=} L \,\&\, \Box^+[R]\Box^+$ (i.e. R contained in E)
c. $\textsc{Perf}_o(L) \overset{\text{def}}{=} L\ \Box^*[R]$ (i.e. $E < R$)

As a position from which to view the event represented by L, R says in the case of $\textsc{Simp}(L)$, that the event has reached completion; in the case of $\textsc{Prog}_o(L)$, that it has *not* quite gotten there yet but is on its way; and in $\textsc{Perf}_o(L)$, that it is history. The subscript o on Perf and on Prog marks the need for additional operations that concern inertia and branching, respectively. We will attend to these complications in sections 3 and 4. For now, we work with truncated forms given by the following definitions. A string $a_1 \ldots a_n$ is *R-truncated* if $R \notin \bigcup_{1\leq i<n} a_i$.

(Thus, $\Box[R]$ is R-truncated, whereas $[R]\Box$ is not. Also, any string where R does *not* occur is R-truncated.) The *R-truncation s_R of* a string s is the longest R-truncated prefix of s; that is,

$$(a_1 \ldots a_n)_R \overset{\text{def}}{=} a_1 \ldots a_k \quad \text{where} \quad k \overset{\text{def}}{=} max\{j \leqslant n | R \notin a_i \text{ for } 1 \leqslant i < j\}.$$

(Thus, $([\mathrm{R}]\Box)_R = [\mathrm{R}]$.) Given a language L, let L_R be the set $\{s_R \mid s \in L\}$ of its R-truncations, and let us say L is *R-truncated* if all its strings are—that is, $L = L_R$. $\textsc{Simp}(L)$ and $\textsc{Perf}_o(L)$ are R-truncated, assuming R does not occur in (L). Not so for $\textsc{Prog}_o(L)$, and accordingly, we define

$$\text{ing}(L) \overset{\text{def}}{=} (\textsc{Prog}_o(L))_R$$

(corresponding to the truncated progressive in Parsons (1990), as opposed to that in Landman (1992) which we reformulate in section 4). We can now take up (3a) and (4a) via the approximations (19) and (20), respectively, suggested by (10a).

(3) a. Pat was walking $\vdash$ Pat walked

(4) b. Pat was walking home $\nvdash$ Pat walked home

(19) $\text{ing}(\mathcal{L}(\text{Pat walk})) \trianglerighteq \textsc{Simp}(\mathcal{L}(\text{Pat walk}))$

(20) $\text{ing}(\mathcal{L}(\text{Pat walk home})) \ntrianglerighteq \textsc{Simp}(\mathcal{L}(\text{Pat walk home}))$

(10) a. $A \vdash B$ because $\mathcal{L}(A) \trianglerighteq \mathcal{L}(B)$

(19) and (20) leave out the speech time S contributed by the past tense in (3a) and (4a). We will see in the next section that our arguments for (19) and (20) remain intact after S is added for (3a) and (4a). An easy route to (19) and (20) is (21).

(21) a. $\mathcal{L}(\text{Pat walk}) = \Box[\text{walk } (p)]^+$
b. $\mathcal{L}(\text{Pat walk home}) = \mathcal{L}(\text{Pat walk})\Box \;\&\; [\overline{\text{home}(p)}]^*[\text{home}(p)]$
$= [\overline{\text{home}(p)}][\text{walk}(p), \overline{\text{home}(p)}]^+[\text{home}(p)]$

Generalizing from (21), what is crucial for (19) is that $\mathcal{L}(\text{Pat walk})$ be divisible, where a language L is *divisible* if $\text{ing}(L) \trianglerighteq L$. (Thus, if L is divisible, then L is iterative and $\text{ing}(L) \trianglerighteq \textsc{Simp}(L)$.) As for (20), observe that

$$\text{ing}(L) \trianglerighteq \Box^*\omega_L \text{ and } L \trianglerighteq \overline{\omega_L}^*\Box \text{ implies } L \nsubseteq nc(\Phi)^*$$

whence

$$\text{ing}(L) \ntrianglerighteq \textsc{Simp}(L) \quad \text{if} \quad L \text{ is telic and } L \subseteq nc(\Phi)^*.$$

Assuming $\mathcal{L}(\text{Pat walk home}) \subseteq nc(\Phi)^*$, the telicity (13a) of $\mathcal{L}(\text{Pat walk home})$ yields (20). Clearly, (19) and (20) hold for various refinements of the languages mentioned there.[3]

In the next section, we consider refinements of languages given by constraints of the form $L \Rightarrow L'$. These constraints are comparable in function to meaning postulates, and typically lead to the addition of fluents. A simple example (injecting a bit of world knowledge) is

$$[\text{dawn}]\square^*[\text{dusk}] \quad \Rightarrow \quad \square^+[\text{noon}]\square^+$$

requiring that any stretch of time that begins with dawn and ends in dusk contains noon. In general, the idea is that for any languages L and L' over the alphabet Pow(Φ), the *constraint* $L \Rightarrow L'$ is the set of strings $s \in \text{Pow}(\Phi)^*$ such that every stretch of s that subsumes L also subsumes L'. The precise notion of stretch is given by that of a factor, where s' is a *factor of* s if $s = us'v$ for some (possibly empty) strings u and v. We then define

$$s \in L \Rightarrow L' \quad \overset{\text{def}}{\Leftrightarrow} \quad \text{for every factor } s' \text{ of } s,\ s' \trianglerighteq L \text{ implies } s' \trianglerighteq L'.$$

How does a constraint $C \subseteq \text{Pow}(\Phi)^*$ refine a language $L \subseteq \text{Pow}(\Phi)^*$? For the case of

$$\begin{aligned} C &= [\text{dawn}]\square^*[\text{dusk}] \Rightarrow \square^+[\text{noon}]\square^+ \\ L &= [\text{rain, dawn}][\text{rain}]^*[\text{rain, dusk}], \end{aligned}$$

a natural refinement of L by C is

$$L_C = [\text{rain, dawn}][\text{rain}]^*[\text{rain, noon}][\text{rain}]^*[\text{rain, dusk}]$$

which filters out the string [rain, dawn][rain, dusk] of length 2 from L, and fleshes out the remaining strings in $[\text{rain, dawn}][\text{rain}]^+[\text{rain, dusk}]$ by inserting noon in the middle. For arbitrary languages L and C over the alphabet Pow(Φ), we form the *application* L_C *of* C *to* L in three steps

(s1) superpose L with $\text{Pow}(\Phi)^*$
(s2) intersect C with the superposition $L \,\&\, \text{Pow}(\Phi)^*$, and
(s3) extract the $\trianglerighteq$-minimal strings of $C \cap (L \,\&\, \text{Pow}(\Phi)^*)$

$$L_C \overset{\text{def}}{=} (C \cap (L\&\text{Pow}(\Phi)^*))_{\trianglerighteq} = (\{s \in C | s \trianglerighteq L\})_{\trianglerighteq}$$

where, in general, $L_{\trianglerighteq}$ is the set of the $\trianglerighteq$-minimal strings in L

[3] A refinement relevant to the progressive has to do with incremental changes in the degree to which a fluent φ holds. We can encode increases in the degree of φ as

$$(\exists x)\ \varphi\text{-deg}(x)\ \wedge (\exists y < x)\ \textit{previous}\ \varphi\text{-deg}(y)$$

using a temporal operator *previous* to shift back one moment in the past.

$$L_{\unrhd} \overset{\text{def}}{=} \{s \in L \mid (\forall s' \in L)\ \ s \unrhd s' \ \text{ implies } \ s = s'\}.$$

Note that L & $\text{Pow}(\Phi)^* \equiv L$, but that (s1) allows us to impose C on L by intersection in (s2). We tidy up in (s3), removing excesses from (s1), as

$$L \ \equiv \ L_{\unrhd} \ \subseteq \ L.$$

Evidently, $L_C \unrhd L$.

3 INERTIA AND FORCE

Turning now to inertia, consider (2a).

(2) a. Pat stopped the car before it hit the tree.

Does (2a) imply that the car hit the tree after Pat stopped it? Not quite. By stopping the car, Pat negates a precondition for $\mathcal{L}$(car hit tree), that precondition being that the car *not* be at rest

$$\mathcal{L}(\text{Pat stop car}) \unrhd \Box^{+}[\text{car-at-rest}] \quad \text{and} \quad \mathcal{L}(\text{car hit tree}) \unrhd \boxed{\text{car-at-rest}}\Box^{+}.$$

$$\underbrace{\cdots[\text{car-at-rest}]}_{\text{Pat stop car}} \quad ??? \quad \underbrace{\boxed{\text{car-at-rest}}\cdots}_{\text{car hit tree}}$$

If the car is to hit the tree, some force must put the car back into motion. In the absence of an intervening force on the stopped car, (2a) suggests that the collision was avoided.

To formalize such reasoning, let us fix a set $\text{Inr} \subseteq \Phi$ of *inertial* fluents such that whenever $\varphi \in \text{Inr}$, $\overline{\varphi} \in \text{Inr}$ and there is a non-inertial fluent $\text{f}\varphi$ marking the application of a force to make φ true at the next moment. We can then express persistence of an inertial fluent φ as the constraint

(per) $$[\varphi]\ \Box \quad \Rightarrow \quad (\Box[\varphi] + [\text{f}\overline{\varphi}]\ \Box)$$

stating that φ persists unless some force is applied to make $\overline{\varphi}$ true (that is, φ false). Forces that make φ true appear in the backward persistence constraint

(pbk) $$\Box[\varphi] \quad \Rightarrow \quad ([\varphi]\Box \ + \ [\text{f}\varphi]\Box)$$

stating that if φ is true, it must have been so previously or else was forced to be true. Differentiating $\text{f}\varphi$ from $\text{f}\overline{\varphi}$ allows us to formulate the constraint 'succeed unless opposed'

(suo) $$[\text{f}\varphi]\Box \ \Rightarrow \ (\Box[\varphi] \ + \ [\text{f}\overline{\varphi}]\Box)$$

saying an unopposed force on φ brings φ about at the next moment. Were $\text{f}\varphi$ and $\text{f}\overline{\varphi}$ the same, (suo) would have no bite (denoting, as it would, the universal language $\text{Pow}(\Phi)^*$). For φ equal to car-at-rest, the collision in (2a) is put into question from two directions: forward from car-at-rest and backward from $\overline{\text{car-at-rest}}$. One direction is enough to block the inference that the car hit the tree—which is fortunate, as we will shortly be forced to modify (per).

Consider again the minimal pair (5), which we can analyse partially via (22), where in(p, d) is an inertial fluent for Pat-in-Dublin and R and S are non-inertial fluents for reference and speech time.

(5) a. Pat left Dublin but is back (in Dublin).
    b. ?Pat has left Dublin but is back (in Dublin).

(22) a. $\mathcal{L}(\text{Pat left Dublin}) \trianglerighteq [\text{in}(p,d)]\Box^*[\text{in}(p,d),\ R]\Box^*[S]$
     b. $\mathcal{L}(\text{Pat has left Dublin}) \trianglerighteq [\text{in}(p,d)]\Box^*[\text{in}(p,d),]\Box^*[R,S]$

To account for the contrast in (5) on the basis of (22), it suffices that in(p,d) persists forward in (22b) but not in (22a).

(23) a. $\mathcal{L}(\text{Pat left Dublin}) \not\trianglerighteq \Box^+[\text{in}(p,d),\ S]$
     b. $\mathcal{L}(\text{Pat has left Dublin}) \trianglerighteq \Box^+[\text{in}(p,d),\ S]$

To derive (23b) from (22b), we need only appeal to (per) and the assumption that in (22b) *no* force is applied against in(p,d) alongside its (displayed) occurrence. By contrast, in (22a)/(23a), it would appear that R blocks in(p,d) from flowing to the right, just as a force against in(p,d) would. Accordingly, we weaken (per) for every inertial φ to

(peR) $$[\varphi]\Box \;\Rightarrow\; (\Box[\varphi] \;+\; [\mathrm{f}\,\overline{\varphi}]\Box \;+\; [R]\Box)$$

adding the possibility $[R]\Box$ to the right-hand side as an alternative to inertial flow $\Box[\varphi]$.[4] Independent confirmation that R is a barrier to forward persistence is provided by (24).

(24) a. Pat was in Dublin $\not\vdash$ Pat is in Dublin
     b. $\mathcal{L}(\text{Pat was in Dublin}) \trianglerighteq [\text{in}(p,d)]^*[\text{in}(p,d),R]\Box^*[S]$

The non-entailment in (24a) suggests that in(p, d) in (24b) does *not* flow to S, as [in(p, d), S] would mean Pat is in Dublin.

[4] Instead of weakening (per) to (peR), it is tempting to assume that R applies a force against all inertial fluents φ occurring alongside it.

(Raf) $$[R,\varphi] \;\Rightarrow\; [\mathrm{f}\,\overline{\varphi}]$$

Unfortunately, (Raf) would wreck our account of the non-veridicality in (2a), assuming R is put at the end of $\mathcal{L}(\text{Pat stop car})$.

A Reichenbachian approach to tense locates S relative to R. Let us write '$R < S$' for the set of strings where R precedes S

$$\text{'}R < S\text{'} \stackrel{\text{def}}{=} \{a_1 \dots a_n \mid (\forall i \text{ such that } R \in a_i) S \notin \bigcup_{1 \leqslant j \leqslant i} a_j\}$$

and '$R = S$' for the set of strings where R and S co-occur

$$\text{'}R = S\text{'} \stackrel{\text{def}}{=} \{a_1 \dots a_n \mid (\forall i)\ R \in a_i \Leftrightarrow\ S \in a_i\}.$$

On languages L where R but not S occurs (e.g. $\textsc{Simp}(L_0)$, $\textsc{Prog}_{\text{O}}(L_0)$, $\textsc{Perf}_{\text{o}}(L_0)$), we can then define the operators

$$\begin{aligned} \textsc{Past}_{\text{o}}(L) &\stackrel{\text{def}}{=} (L\Box^* \ \&\ \Box^+[S]\Box^*) \cap \text{'}R{<}S\text{'} \cap \text{Unpad} \\ \textsc{Pres}(L) &\stackrel{\text{def}}{=} (L \ \&\ \Box^*[S]\Box^*) \cap \text{'}R = S\text{'} \cap \text{Unpad} \end{aligned}$$

where Unpad consists of non-empty strings that neither begin nor end with $\Box$

$$\text{Unpad} \stackrel{\text{def}}{=} \text{Pow}(\Phi)^+ - (\Box\text{Pow}(\Phi)^* + \text{Pow}(\Phi)^*\Box).$$

Then, for instance,

$$\begin{aligned} \mathcal{L}_{\text{o}}(\text{Pat left Dublin}) &= \textsc{Past}_{\text{o}}(\textsc{Simp}(L_0)) \\ \mathcal{L}_{\text{o}}(\text{Pat has left Dublin}) &= \textsc{Pres}(\textsc{Perf}_{\text{o}}(L_0)) \end{aligned}$$

where L_0 is $\mathcal{L}$(Pat leave Dublin), and the subscripts o indicate that adjustments may be necessary to satisfy constraints involving, for instance, inertia.

These constraints are hard inasmuch as the implications $L_1 \Rightarrow L_2$ are non-defeasible. Indeed, (2a) non-defeasibly implies (25).

(2) a. Pat stopped the car before it hit the tree.

(25) Unless some force put the stopped car back in motion, the car did not hit the tree.

Defeasibility in inertia creeps in not through $\Rightarrow$ but through the non-determinism + to the right of $\Rightarrow$— for example, in the choice between $[\varphi]\Box$ and $[\mathrm{f}\varphi]\Box$ in

(pbk) $$\Box[\varphi] \ \Rightarrow \ ([\varphi]\Box \ + \ [\mathrm{f}\varphi]\Box).$$

To assume that in a language L, no forces apply except those explicitly appearing in L is to restrict the application L_C of the inertial constraints C to L as follows. In step (s1) for L_C, we restrict the fluents added to the set Inr of inertial fluents, superposing L with $\text{Pow}(\text{Inr})^*$, rather than all of Pow(Φ)*. If we then carry out steps (s2) and (s3), the end result refines L_C to

$$L^{\mathrm{i}}_C \overset{\mathrm{def}}{=} (C \cap (L \mathbin{\&} \mathrm{Pow}(\mathrm{Inr})^*))_{\unrhd}$$

(with i standing for inertial). Clearly, $L^{\mathrm{i}}_C \subseteq L_C$, the difference being that L^{i}_C adds only inertial fluents (in Inr) to L, whereas L_C may introduce forces into L. There are, of course, cases where inertial flow is undesirable, as in (26), which presumably does *not* entail *Pat is in Dublin*.

(26) Pat has been in Dublin.
$[\mathrm{in}(p,d)]^*[\mathrm{in}(p,d)]\Box^*[R,S]$

To block in(p, d) from flowing into the box $[R, S]$ in (26), it suffices to add a force fluent freezing in(p, d), so that

$$\mathcal{L}(\text{Pat has been in Dublin}) \ \unrhd\ [\mathrm{in}(p,d)]^*[\mathrm{in}(p,d), \mathrm{f}\overline{\mathrm{in}(p,d)}]\Box^*[R,S].$$

In general, we can block inertial flow (forward) of L's postconditions in $L\Box^*[R]$ by turning L into $\{s^\bullet | s \in L\}$, where $s^\bullet$ is s with all the necessary force fluents $\mathrm{f}\overline{\varphi}$ at the end

$$s^\bullet \overset{\mathrm{def}}{=} s \ \& \ \Box^*[\mathrm{f}\overline{\varphi}|\varphi \in \mathrm{Inr} \ \text{and} \ s \unrhd \Box^*[\varphi]]$$

(that is, $(sa)^\bullet$ is $s(a \cup [\mathrm{f}\overline{\varphi}|\varphi \in \mathrm{Inr} \cap \mathrm{a}])$). It may be the case that for the perfect operator to apply to a language L, some forces must be at work in L.

So why in (2a) are we so reluctant to add forces enabling the car to hit the tree?

(2) a. Pat stopped the car before it hit the tree.

Perhaps because, as we will see in the next section, we are not forced to; we may branch.

4 BRANCHING AND COHERENCE

Whether or not the most natural reading of (2a) implies (2b), the question arises: How are we to interpret (2b)?

(2) b. The car did not hit the tree but it might well have.

Under dynamic semantics, the sentence *the car did not hit the tree* eliminates all epistemic possibilities where the car hit the tree, making the clause *the car might have hit the tree* unacceptable (Veltman 1996). This is all well and good if *might* is read epistemically as in (2c).

(2) c. The car did not hit the tree but ?for all we know, it might have.

There is, however, a metaphysical reading of *might*, under which (2b) is perfectly acceptable (Condoravdi 2002).

(2) d. Had Pat not stopped the car, it might well have hit the tree.

To make sense of *might* in (2d), let us assume that we can turn a string s into a fluent may(s) saying roughly that from the next moment on, s may be observed. We will presently do away with such fluents may(s), but for now, they are convenient for making the following point. In (27), if the string s_1 depicts *car stop* and s_2 depicts *car hit tree*, then s_3 depicts *car stop but may have hit tree*.

(27) a. $s_1 \overset{\text{def}}{=\!=} \overline{[\text{car-at-rest}]}\,[\text{car-at-rest}]$
b. $s_2 \overset{\text{def}}{=\!=} [\overline{\text{car-at-rest},}\ \overline{\text{car-tree-contact}}][\text{car-tree-contact}]$
c. $s_3 \overset{\text{def}}{=\!=} \overline{[\text{car-at-rest}],}\,\text{may}(s_2)]\ [\text{car-at-rest}]$

Whereas information in Veltman (1996) grows solely by elimination, we not only have

$$L \subseteq L' \quad \text{implies} \quad L \trianglerighteq L'$$

but also expansive growth such as

$$s_3 \ \trianglerighteq\ s_1$$

in (27). Veltman's *might* depends on the full epistemic state for its interpretation (and is in this sense said to be non-distributive); the fluent may(s) is interpreted pointwise at each possibility in the epistemic state. Up to now, the points in our epistemic states have been strings. To interpret may(s), we extend strings to branching strings formed from an alphabet not only by concatentation, with

$$\text{ss}' \quad \text{saying} \quad \text{s}' \ \text{follows s},$$

but also by a second binary operation β, with

$$\beta(\text{s,s}') \quad \text{saying} \quad \text{s}' \ \text{may follow s}.$$

We should be careful *not* to confuse non-deterministic choice + with β. Epistemic uncertainty is encoded by + through a *set* of strings (branching and otherwise), whereas β encodes metaphysical non-determinism within a branching string. (Introducing metaphysical possibilities via β adds information insofar as it eliminates epistemic possibilities in which the branches are ruled out as metaphysical possibilities.) To illustrate, we can flesh out s_3 in (27) as the branching string

$$\hat{s} \overset{\text{def}}{=} \beta([\overline{\text{car-art-rest}}], s_2)[\text{car-at-rest}].$$

In $\hat{s}$, the first snapshot $[\overline{\text{car-at-rest}}]$ is followed by [car-at-rest], which clashes with the first snapshot of s_2 (on whether or not the car is at rest), making s_2 a counterfactual continuation of $[\overline{\text{car-at-rest}}]$.[5] We can read (28) off $\hat{s}$ inasmuch as chopping off the suffix [car-a-rest] from $\hat{s}$ restores s_2 as a live option after $[\overline{\text{car-at-rest}}]$.

(28) Had it not stopped, the car may have hit the tree.

What is the general mechanism for forming $\hat{s}$ from s_1 and s_2? The mechanism is a generalization $\&_\Sigma$ of superposition & based on a restriction $\Sigma \subseteq \text{Pow}(\Phi)$ of the alphabet $\text{Pow}(\Phi)$ to a fixed set Σ of 'legal' snapshots. For $\Sigma = \text{Pow}(\Phi)$, $\&_\Sigma$ is just &. But the idea behind throwing out from Σ a snapshot such as

$$[\text{car-at-rest}, \ \overline{\text{car-at-rest}}] \ = \ [\text{car-at-rest}] \cup [\overline{\text{car-at-rest}}]$$

is that it contains clashing parts [car-at-rest] and $[\overline{\text{car-at-rest}}]$ that ought not to be union-ed, but must be kept apart, hence the branching. As with superposition &, it suffices to combine strings of the same length (padding them with empty boxes $\Box$ at either end, if necessary). An example with Σ equal to the set $nc(\Phi)$ of non-contradictory snapshots is (29), where $\hat{s}$, s_1 and s_2 are as in (27).

$$\begin{aligned}
(29) \quad \hat{s}\Box \ &= \ \beta([\overline{\text{car-at-rest}}], s_2) \ \ [\text{car-at-rest}]\Box \\
&= \ s_1\Box \quad \&_{nc(\Phi)} \quad \Box s_2 \\
&= \ [\overline{\text{car-at-rest}}][\text{car-at-rest}]\Box \ \ \&_{nc(\Phi)} \\
&\quad \ \Box[\overline{\text{car-at-rest}}, \ \overline{\text{car-tree-contact}}][\text{car-tree-contact}]
\end{aligned}$$

In general, we form $s \ \&_\Sigma s'$ from strings $s = a_1 \ldots a_n$ and $s' = a'_1 \ldots a'_n$ that are both in Σ^n for some fixed n, and let k be the largest $i \leqslant n$ such that $(a_1 \cup a'_1) \ldots (a_i \cup a'_i) \in \Sigma^*$—that is,

$$k \ \overset{\text{def}}{=} \ \max\{i \leqslant n | a_j \cup a'_j \in \Sigma \ \text{ for } \ 1 \leqslant j \leqslant i\}.$$

[5] An alternative to $\hat{s}$ (suggested by Stefan Kaufmann) is the branching string

$$\dot{s} \overset{\text{def}}{=} \beta([\overline{\text{car-at-rest}}, \overline{\text{car-tree-contact}}], s_1)[\text{car-tree-contact}].$$

obtained from the string s_4 in (27d) by unwinding may(s_1).

(27) d. $s_4 \overset{\text{def}}{=} [\overline{\text{car-at-rest}}, \ \overline{\text{car-tree-contact}}, \text{may}(s_1)][\text{car-tree-contact}]$.

According to $\dot{s}/s_4$, the car hit the tree (s_2) but may have instead stopped (s_1).

(In the case of (29), $s = s_1\square$, $s' = \square s_2$, $n = 3$ and $k = 1$.) The Σ-*zipper* $s \,\&_\Sigma s'$ is defined iff $k \geqslant 1$, in which case $s \,\&_\Sigma s'$ preserves all of s and as much of s' as s and Σ allow, the remainder of s' attaching to s via β. That is, if $k \geqslant 1$, let

(i) s'' be the superposition $a_1 \ldots a_k \;\&\; a'_1 \ldots a'_k$ of the k-length prefixes of s and s'

$$s'' \overset{\text{def}}{=\!=} (a_1 \cup a'_1) \ldots (a_k \cup a'_k)$$

(so that $s'' = s \,\& s'$ if $k = n$)

(ii) s_+ and s'_+ be the suffixes of s and s' from positions $k + 1$ to n

$$s_+ \overset{\text{def}}{=\!=} a_{k+1} \ldots a_n$$
$$s'_+ \overset{\text{def}}{=\!=} a'_{k+1} \ldots a'_n$$

so that $s = a_1 \ldots a_k s_+$ and $s' = a'_1 \ldots a'_k s'_+$ (e.g. in (29), s_+ is [car-at-rest]$\square$ while s'_+ is [car-at-rest, car-tree-contact][car-tree-contact])

(iii) $s \,\&_\Sigma s'$ be s'' plus any remaining parts s_+ and s'_+ attached by concatenation and β, respectively,

$$s \,\&_\Sigma s' \overset{\text{def}}{=\!=} \begin{cases} s'' & \text{if } k = n \\ \beta(s'', s'_+)s_+ & \text{otherwise} \end{cases}$$

(whence (29) for $\Sigma = nc(\Phi)$).

The requirement $k \geqslant 1$ ensures that s' can attach to s, with $s \,\&_\Sigma s'$ maximizing the attachment of s' to s. Next, given languages $L, L' \subseteq \Sigma^*$, we collect Σ-zippers from their strings in the Σ-*zipper*

$$L \,\&_\Sigma\, L' \overset{\text{def}}{=\!=} \bigcup_{n \geqslant 1} \{a_1 \ldots a_n \,\&_\Sigma\, a'_1 \ldots a'_n | a_1 \ldots a_n \in L \text{ and } a'_1 \ldots a'_n \in L' \text{ with } a_1 \cup a'_1 \in \Sigma\}.$$

The ordinary (non-branching) strings in $L \,\&_\Sigma L'$ are the strings in Σ^* belonging to the ordinary superposition $L \,\&\, L'$.

(30) $\qquad (L \,\&_\Sigma\, L') \cap \mathrm{Pow}(\Phi)^* = (L \,\&\, L') \cap \Sigma^*$

The remainder $(L \,\&_\Sigma L') - (L \,\&\, L')$ represents cases of branching and counterfactual continuations, on which we might impose additional requirements (beyond snapshots being from Σ).

Going back to *before*, the idea roughly is to represent a veridical reading of A *before* B, in which both arguments A and B are asserted to hold, by *certain* non-branching strings from (30) where $L = \square^*\mathcal{L}(A)\square^*$ and $L' = \square^*\mathcal{L}(B)\square^*$.(We say *certain* because such strings must satisfy additional constraints; e.g. an occurrence of an *A*-string must precede that of a *B*-string.) In a counterfactual reading where *B* is asserted not to hold, the strings branch.[6] For concreteness, let us focus on the branching reading of (2a) in which the car does not hit the tree. As a representative branching string for that reading, $\hat{s}$ (i.e. $\beta([\overline{\text{car-at-rest}}], s_2)[\text{car-at-rest}])$) suggests that (2a) implies (31a).

(2) a. Pat stopped the car before it hit the tree.

(31) a. Before Pat stopped the car, it may have been the case that the car was on course to hit the tree.
b. Before Pat stopped the car, it was going to hit the tree.

Indeed, depending on what we make of the phrase *going to*, we might extract (31b) from the following modification of $\hat{s}$. Let us add the snapshot

$$a \overset{\text{def}}{=} [\overline{\text{car-at-rest}}, \overline{\text{car-tree-contact}}]$$

to s_2, deriving (as it were) as_2 from $\square s_2$ by backward inertial flow (pbk).

(pbk) $\square[\varphi] \quad \Rightarrow \quad ([\varphi]\square \; + \; [\mathrm{f}\varphi]\square)$

We then sharpen (29) to (32).

(32) $s_1\square \;\; \&_\Sigma \;\; as_2 = \beta(a, s_2)[\text{car-at-rest}]\square$

In (32), s_1 depicts *car stop*, while as_2 depicts *car hit tree*. The transition in s_1 from $[\overline{\text{car-at-rest}}]$ to [car-at-rest] requires a force (assuming car-at-rest is inertial) that in (32) gives rise to a branch marking *car hit tree* counterfactual. It is easy to see how in general branching from the zipper arises from a force disturbing inertia: *no* force, no change in inertial fluents. Furthermore, a force leads to branching only if it is opposed. In (33), Pat does *not* contribute a force to stop the car; therefore, it is natural to read (33) as entailing the car hit the tree.

(33) Pat did not stop the car before it hit the tree.

Next, consider (34) and (35).

[6] Mixing non-branching and branching strings, we can represent the non-committal readings in Beaver & Condoravdi (2003) of *A before B* where *B* may or may not hold. That is, non-committal readings arise from epistemic uncertainty (i.e. non-deterministic choice +) between veridical and counterfactual ones. More below.

(34) The car was at rest before it hit the tree.
(35) The car came to rest before it hit the tree.

How can we account for reading *before* veridically in (34) but counterfactually in (35)? The *car-hit-tree* string as_2 cannot attach anywhere where the car is at rest (in (34)), but can attach before the car comes to rest (in (35)). We arrive at constraint (36), where $before_c$ and $before_v$ are the counterfactual and veridical forms, respectively, of *before*.

(36) $A \ before_c \, B$ implies $going\text{-}to(B) \ before_v A$

To satisfy (36), we have fleshed out $\Box s_2$ to as_2, under the assumption that $going\text{-}to(car\text{-}hit\text{-}tree)$ implies $\overline{\text{car-at-rest}}$ and $\overline{\text{car-tree-contact}}$. But just what does $going\text{-}to(B)$ mean?

To bring out the non-veridicality of the progressive, recall the aspectual operators SIMP, PROG_o and PERF_o in (18) employ the reference time R to mark what has come to pass from what is yet to come. We can specify branching at R through a function κ: $\text{Pow}(\Phi)^+ \to \text{Pow}(\text{Pow}(\Phi)^+)$ mapping a string $s \in \text{Pow}(\Phi)^+$ to a set $\kappa(s)$ of strings that are identical with s up to R

(c1) $(\forall s' \in \kappa(s)) s'_R = s_R$

(where s_R is the R-truncation of s as defined in section 2) and includes s

(c2) $s \in \kappa(s)$.

It is useful to think of strings in $\kappa(s)$ as R-continuations of s, and to expand a language L into the language L_κ of R-continuations of strings in L

$$L_\kappa \overset{\text{def}}{=} \cup \{\kappa(s) | s \in L\}.$$

Clearly, if κ maps every $s \in \Sigma^+$ to $\{s\}$, then $L_\kappa = L$. But more generally, from (c2), we may conclude that L_κ contains L

$$L \subseteq L_\kappa$$

and from (c1), that L and L_κ have the same R-truncations

$$L_R = (L_\kappa)_R.$$

In particular, $\text{SIMP}(L) = (\text{SIMP}(L)_\kappa)_R$ and $\text{PERF}_o(L) = (\text{PERF}_o(L)_\kappa)_R$ for languages L where R does not occur. By contrast, $\text{PROG}_o(L)$ may differ from $(\text{PROG}_o(L)_\kappa)_R$ if $\kappa(s) \neq (s)$, with alternatives to s encoded in $\kappa(s) - \{s\}$. The crucial point is that R is a realis marker: beyond R, events may develop in as many different ways as κ allows.

The presentation of the language L_κ above is designed to bring out the similarity with the interpretation of metaphysical *might* in Condoravdi (2002). Condoravdi imposes a condition of *historical necessity* (Thomason 1984) on the metaphysical modal base, which is expressed above as (c1). To link L_κ to branching strings, it is convenient to alter the presentation slightly. Given a string s, let s^R be the suffix of s cut off by the R-truncation s_R of s in that

$$s = s_R \ s^R.$$

(For instance, $(\Box[R][\varphi])^R = [\varphi]$.) Next, let us collect the alternatives to s^R specified by $\kappa(s)$ in

$$\kappa^R(s) \overset{\text{def}}{=} \{s'^R | s' \in \kappa(s) - \{s\}\}.$$

Then,

$$L_\kappa \quad = \quad L \ \cup \ \{s_R s' | s \in L \ \text{ and } \ s' \in \kappa^R(s)\}$$

and if say, $\kappa^R(s) = \{s', s''\}$ we can attach s' and s'' to s via β, encoding $\kappa(s)$ as the branching string

$$\beta(\beta(s_R, s'), s'') \ \ s^R.$$

This encoding generalizes to strings s and functions κ such that $\kappa(s)$ is finite.[7] Conversely, from say, the branching string

$$\beta([R], s')\beta(a, s'') \ \ b$$

we can get three R-continuations of $[R]ab$

$$\{[R]s', \ [R]as'', \ [R]ab\} \quad \subseteq \quad \kappa([R]ab).$$

The set of R-continuations obtained from a branching string is partial in the same sense that the snapshots are partial.

What can we say about R-continuations? Can we, for instance, assume (37), which assigns s and s' the same R-continuations whenever they are identical up to R?

[7] The order in which $\kappa^R(s)$ is enumerated is immaterial, assuming we impose the following equations between branching strings

$$\begin{aligned} \beta(\beta(s, s'), s') \quad &= \quad \beta(s, s') \\ \beta(\beta(s, s'), s'') \quad &= \quad \beta(\beta(s, s''), s') \end{aligned}$$

not to mention, $s\beta(s', s'') = \beta(ss', s'')$ and $(ss')s'' = s(s's'')$. More in Fernando (2005).

(37) If $s_R = s'_R$ then $\kappa(s) = \kappa(s')$.

If (37) were true then by (c2), every string identical to s up to R must be in $\kappa(s)$

$$s_R = s'_R \quad \text{implies} \quad s' \in \kappa(s') = \kappa(s)$$

and so by (c1),

$$\kappa(s) \quad = \quad \{s' | s_R = s'_R\}.$$

We must reject (37) if we want more restricted sets $\kappa(s)$. (38) is a restriction corresponding to the *normality condition* in Beaver and Condoravdi (2003).

(38) $\kappa(s)$ consists only of strings that are reasonably probable given what has happened up to R.

An argument against (38) is provided by the following passage from Landman (1992).

> if an accomplishment manages to get completed, it is unproblematic to assume (in retrospect) that the progressive is true during the development stage ... even if the event gets completed against all odds. (p. 14)

It would seem that if we are to restrict $\kappa(s)$ to the 'reasonably probable' continuations, then that probability judgment had better be made on the basis of events not only up to R but also beyond R, contra (38). But then how are we to keep that probability judgment from degenerating to an assignment of 0 to counterfactual events and 1 to actual events? It is, of course, easy enough to replace $\kappa(s)$ in (38) by $\kappa(s) - \{s\}$. But it is even easier to put (38) and probabilities aside, and try make do with (36).

(36) $A \ \mathit{before}_c B \quad \text{implies} \quad \mathit{going\text{-}to}(B)\ \mathit{before}_v A$

There is a vagueness in what *go-to*(B) means that is difficult to tease out from the progressive. Be that as it may, we can apply (36) to fix a problem pointed out in Beaver & Condoravdi (2003) for the 'Anscombe/Landman analysis' of *before*:

> Even if David never won a gold medal at anything in his whole life, provided he ate lots of ketchup, (x) is predicted true.
>
> (x) David ate lots of ketchup before he made a clean sweep of all the gold medals in the Sydney Olympics.

Assuming *before* in (x) is counterfactual, (36) and (x) yield (39) with *before* veridical.

(39) Before David ate lots of ketchup, he was going to make a clean sweep of all the gold medals in the Sydney Olympics.

(x) is rejected because no string describing David winning an Olympic medal attaches via β to a factual string.

Does (36) hold if $before_c$ in its left-hand side is replaced by $before_v$? (40) suggests perhaps not.

(40) a. Columbus discovered America before the Soviet Union collapsed.
b. ?Before Columbus discovered America, the Soviet Union was going to collapse.

Evidently, there are limits to how far back *going-to*(B) can stretch. But then, (36) was motivated above by branching from a zipper—apologies for mixing metaphors—and not by the (veridical) case of a zipper coinciding with superposition. (What (36) adds to the zipper is illustrated above by the step from (29) to (32).)

We have defined Σ-zippers to be as close to superposition as Σ allows, zipping zippers as high up as we could without straying from Σ. This is in line with the constraint *Maximise Discourse Coherence* (MDC) in Asher & Lascarides (2003) if we agree that discourse coherence increases as the zipper is zipped (bringing the two strings closer together). Just as epistemic uncertainty (represented in a language as non-deterministic choice + between its possibilities/strings) is subject to entailment, indeterminism expressed by branching (β) is subject to coherence. A further constraint one might apply to counterfactual *before*, suggested by the *initial branch point condition* in Beaver & Condoravdi (2003), is to sharpen the veridical *before* in the right-hand side of (36) to *one moment before*, strengthening (31b) to (31c).

(41) A $before_c$ B implies *going-to* (B) $one\text{-}moment\text{-}before_v A$

(31) b. Before Pat stopped the car, it was going to hit the tree.
c. Up until Pat stopped the car, it was going to hit the tree.

Translating (41) into an English test is perhaps more awkward than is the case for (36). At any rate, it has a precise enough meaning for our discrete branching string representations: in the zipper representing A $before_c$ B, an A-string is formed by position $k + 1$, where k is as high up as the zipper zips up. (It is tempting to view (41) as an instance of MDC for counterfactual readings; any higher up and we get a veridical reading.) The discreteness here of time reflects the temporal granularity of the conceptualized event, and more generally, the perspectival character of branching. What *one moment before* and *going-to*(B) mean are arguably matters of perspective, not metaphysics. Similarly, the *well* in (2b) reflects how picky the continuations $\kappa(s)$ of a string s might be (one moment) before Pat stopped the car.

(2) a. Pat stopped the car before it hit the tree.
 b. The car did not hit the tree but it might well have.

5 CONCLUSION

Let us sum up. Events were formulated as strings over an alphabet consisting of sets of fluents in section 2. These strings were then extended along two dimensions, with an eye to Reichenbachian treatments of the perfect and the progressive. In section 3, strings were extended to the right (to attach the reference time for the perfect), subject to inertial constraints. And in section 4, they were extended sideways to accommodate branching beyond the reference time (for the progressive). We can extend strings lengthwise and sideways indefinitely to obtain, at the limit, worlds from extensions that satisfy appropriate constraints such as Φ-bivalence

$$\cap_{\varphi\in\Phi} \square \quad \Rightarrow \quad ([\varphi]+[\bar{\varphi}])$$

(deciding at each moment whether φ holds or not) and Φ-non-contradictoriness

$$nc(\Phi)^* \quad = \quad \bigcap_{\varphi\in\Phi}([\varphi,\bar{\varphi}] \quad \Rightarrow \quad \emptyset)$$

(ruling out co-occurrences of φ and $\bar{\varphi}$). But we need not proceed to the limit to examine entailments. Sidestepping worlds, we can work instead with superposition &, subsumption $\trianglerighteq$, $\Rightarrow$-constraints and zippers $\&_\Sigma$.

The question remains: what does all this have to say about Dowty's intuition that the progressive of an event is true at world w and time I if the event culminates in (I, w)-inertia worlds? Within our framework, an event culminates in a world containing a string representing the event. Attaching Reichenbach's reference time R to the right end point of the interval I, we allow branching at R so that the progressive of an event might be true in a world where that event does not culminate. Can we extract from this a concept of an (I, w)-inertia world where every event in progress at (I, w) is guaranteed to culminate? Not quite.

Consider the strings $ab[\varphi]$ and $ab[\bar{\varphi}]$ where $R \in b$ and $a, b \in nc(\Phi)$. We can combine the two strings to form the branching string

$$ab[\varphi] \;\&_{nc(\Phi)}\; ab[\bar{\varphi}] \quad = \quad a\beta(b, [\bar{\varphi}])[\varphi]$$

or

$$ab[\bar{\varphi}] \;\&_{nc(\Phi)}\; ab[\varphi] \quad = \quad a\beta(b, [\varphi])[\bar{\varphi}]$$

or $a\beta(\beta(b, [\bar{\varphi}]), [\varphi])$, but not the string $ab[\varphi, \bar{\varphi}] \notin nc(\Phi)^*$, as φ clashes with $\bar{\varphi}$. In other words, two different events may be partially realized at (I, w) even though there is no single world where they can both culminate. To guarantee the culmination of an event e, it seems we must relativize the notion of an inertia world not to (I, w) but to e.[8] Whether or not this trivializes Dowty's intuition, it is hardly surprising then that the bulk of this paper should concern not worlds but events. These events are shaped by perspectives specifying constraints that strings representing events must satisfy. The strings may, under pressure from other strings, branch due to forces that compete to overturn inertia.[9]

Acknowledgements

This paper grew out of a talk titled 'What is an inertia world?' presented at the *Language under Uncertainty* workshop in Kyoto University, 21–23 January 2005. I thank the organizers and participants of this workshop and the editors of this special issue, Takao Gunji, Stefan Kaufmann and Yukinori Takubo. I am indebted to two anonymous referees as well as Stefan Kaufmann for detailed and penetrating comments that resulted in substantial revisions.

TIM FERNANDO
Computer Science Department
Trinity College
Dublin 2
Ireland
e-mail: tim.fernando@tcd.ie

[8] The alternative (mentioned in section 1) is to insist that *no* two events headed for a conflict can be in progress at the same pair (I, w). To pursue this alternative, we can impose further constraints (on the realization of the progressive) to that effect. I leave a detailed development of this for some other occasion.

[9] Behind these forces lurk automata, which I have left out above but explore elsewhere (Fernando 2005, 2006).

REFERENCES

Asher, N. & Lascarides, A. (2003), *Logics of Conversation*. Cambridge University Press. Cambridge, MA.

Beaver, D. & Condoravdi, C. (2003), 'A uniform analysis of *before* and *after*'. In R. Young and Y. Zhou (eds.), *Proceedings of Semantics and Linguistic Theory XIII*. Cornell Linguistics Circle Publications. Ithaca, NY. 37–54.

Condoravdi, C. (2002), 'Temporal interpretation of modals: modals for the present and for the past'. In D. Beaver, S. Kaufmann, B. Clark and L. Casillas (eds.), *The Construction of Meaning*. CSLI. Stanford, CA. 59–88.

Dowty, D. R. (1979), *Word Meaning and Montague Grammar*. Reidel. Dordrecht.

Fernando, T. (2004), 'A finite-state approach to events in natural language semantics'. *Journal of Logic and Computation*, **14**:79–92.

Fernando, T. (2005), 'Events from temporal logic to regular languages with branching'. In G. Jaeger, P. Monachesi, G. Penn, J. Rogers and S. Winter (eds.), *Proceedings of 10th Formal Grammar and 9th Mathematics of Language*. Stanford, Edinburgh. 27–38.

Fernando, T. (2006), 'Finite-state temporal projection'. In O. Ibarra and H.-C. Yen (eds.), *Proceedings of 11th International Conference on Implementation and Application of Automata (CIAA 2006)*. Lectures Notes in Computer Science 4094. Springer-Verlag, Berlin. 230–41.

van Lambalgen, M. & Hamm, F. (2004), *The Proper Treatment of Events*. Blackwell. Oxford.

Landman, F. (1992), 'The progressive'. *Natural Language Semantics* **1**:1–32.

McCarthy, J. & Hayes, P. (1969), 'Some philosophical problems from the standpoint of artificial intelligence'. In M. Meltzer and D. Michie (eds.), *Machine Intelligence 4*. Edinburgh University Press. Edinburgh. 463–502.

Moens, M. & Steedman, M. (1988), 'Temporal ontology and temporal reference'. *Computational Linguistics* **14**:15–28.

Parsons, T. (1990), *Events in the Semantics of English: A Study in Subatomic Semantics*. MIT Press. Cambridge, MA.

Reichenbach, H. (1947), *Elements of Symbolic Logic*. London. Macmillan.

Steedman, M. (2000), The Productions of Time. Draft, ftp://ftp.cogsci.ed.ac.uk/pub/steedman/temporality/temporality.ps.gz. An update of 'Temporality'. In J. van Benthem and A. ter Meulen (eds.), *Handbook of Logic and Language*. Elsevier. North Holland. 895–935.

Tenny, C. (1987), *Grammaticalizing Aspect and Affectedness*. Dissertation, Department of Linguistics and Philosophy, MIT. Cambridge, MA.

Thomason, R. (1984), 'Combinations of tense and modality'. In D. Gabbay and F. Guenthner (eds.), *Handbook of Philosophical Logic*. Reidel. 135–65.

Veltman, F. (1996), 'Defaults in update semantics'. *Journal of Philosophical Logic* **25**:221–61.

Vendler, Z. (1967), *Linguistics in Philosophy*. Cornell University Press. Ithaca, NY.

First version received: 10.10.2005
Second version received: 02.10.2006
Accepted: 01.04.2007