Natural Language Engineering **17** (2): 163–184.
doi:10.1017/S135132491100009X

Regular relations for temporal propositions

TIM FERNANDO
Trinity College Dublin, Ireland
e-mail: tim.fernando@cs.tcd.ie

(*Received 14 June 2010; revised 22 September 2010; accepted 22 October 2010*)

Abstract

Relations computed by finite-state transducers are applied to interpret temporal propositions in terms of strings representing finite contexts or situations. Carnap–Montague intensions mapping indices to extensions are reformulated as relations between strings that can serve as indices and extensions alike. Strings are related according to information content, temporal span and granularity, the bounds on which reflect the partiality of natural language statements. That partiality shapes not only strings-as-extensions (indicating what statements are about) but also strings-as-indices (underlying truth conditions).

1 Introduction

There is a substantial body of work applying automata theory to temporal reasoning, with strings (and trees) stretching to infinity to describe non-terminating computation runs (e.g. Emerson 1992; Vardi 2007). Many natural language statements, however, describe temporally bounded events, including reports in the past tense, such as *Pat walked to the post office* or *An event of such and such happened*, that ensure the events have an end (if not a beginning). The question arises: can we employ ordinary finite-state methods (of the sort in, for example, Beesley and Karttunen 2003) to reason about strings that can be assumed finite? The main aim of this paper is to show how we might, and how that may stray from established practices. The focus is not on any particular natural language engineering application, but rather on basic conceptual issues that bear on such applications. These issues revolve around events and situations natural language statements are about—events and situations bounded in ways that merit investigation.

1.1 Linear temporal logic and finite situations

For orientation, we start with linear temporal logic (LTL) over the set $\mathbb{Z}$ of integers and a set P of atomic formulas (e.g. Vardi 2007). LTL-formulas are evaluated against relations $V \subseteq \mathbb{Z} \times P$ between $\mathbb{Z}$ and P, with $V(t,p)$ read 'p is true at V,t' and written $V,t \models p$

$$V,t \models p \iff V(t,p)$$

for $t \in \mathbb{Z}$ and $p \in P$. Fixing the evaluation time to 0, we can describe an infinite string of p's through the LTL-formula $\mathsf{G}p$ that asserts $V(n,p)$ for every non-negative integer n

$$\begin{aligned} V,0 \models \mathsf{G}p &\iff (\forall n \geq 0)\ V(n,p) \\ &\iff \{(n,p) \mid n \geq 0\} \subseteq V. \end{aligned}$$

Negative integers come in to interpret past operators such as *previous*, from which we can form the LTL-formula $\mathit{previous}(q) \vee \mathsf{G}p$ that is true at $V,0$ iff $\mathsf{G}(p)$ is or q is true at $V,-1$

$$\begin{aligned} V,0 \models \mathit{previous}(q) \vee \mathsf{G}p &\iff V(-1,q) \text{ or } (\forall n \geq 0)\ V(n,p) \\ &\iff \{(-1,q)\} \subseteq V \text{ or } \{(n,p) \mid n \geq 0\} \subseteq V \end{aligned}$$

(for $q \in P$). Assuming P is closed under complementation,[1] one can, in general, map an LTL-formula φ and integer t to a set $\mathbf{S}[\varphi,t]$ of relations $s \subseteq \mathbb{Z} \times P$, reducing the truth of φ at V,t to some subset of V being in $\mathbf{S}[\varphi,t]$

$$V,t \models \varphi \iff (\exists s \in \mathbf{S}[\varphi,t])\ s \subseteq V \tag{1}$$

(e.g. Fernando 2009a). For instance, $\mathbf{S}[\mathit{previous}(q) \vee \mathsf{G}p, 0]$ consists of the two relations $\{(-1,q)\}$ and $\{(n,p) \mid n \geq 0\}$, each describing a way for $\mathit{previous}(q) \vee \mathsf{G}p$ to be true at 0. As this example shows, a relation s in $\mathbf{S}[\varphi,t]$ may or may not be finite.

Of course, if V were finite, then we can throw out all infinite relations from $\mathbf{S}[\varphi,t]$. Put another way, we can restrict $\mathbf{S}[\varphi,t]$ to finite relations by focusing on finite subsets of V (reconstructing V as the union of its finite subsets, as in the reformulation of Büchi acceptance in terms of paths in Fernando 2008b). But we had better be careful about asserting line (1) above for different temporal spans (finite or not), as made clear by the formula $\mathsf{G}p$ saying 'p now and forever more' within that span. To establish (1) for spans over only the two times 0 and 1, we put

$$\mathbf{S}[\mathsf{G}p, 0] = \{\{(0,p),(1,p)\}\}$$

but if we add 2 to that span, we would require

$$\mathbf{S}[\mathsf{G}p, 0] = \{\{(0,p),(1,p),(2,p)\}\}.$$

Evidently, varying the temporal spans under consideration (from $\mathbb{Z}$ to its finite segments) renders (1) untenable—at least for φ's such as $\mathsf{G}p$. Such complications beg the question: apart from sidestepping infinite relations $s \in \mathbf{S}[\varphi,t]$, why should we cut an infinite relation $V \subseteq \mathbb{Z} \times P$ down to its finite parts?

At the philosophical end, there is something to be said for grounding semantics in our finite experience, minimizing mysterious abstractions such as possible worlds by working instead with situations that are partial (e.g. Barwise and Perry 1983). From a more practical natural language processing perspective, it is far from clear that one notion of time will do. Temporal spans vary, as does temporal granularity. *Every Monday* might mean every Monday in June or every Monday in 2010, and *the*

[1] That is, negation is pushed down to the level of atomic formulas using De Morgan's laws and counter-extensions.

previous moment might refer to the previous minute or previous second (among many other possibilities). The analysis of calendar expressions in Niemi and Koskenniemi (2009) is particularly instructive for our purposes.

1.2 Calendar expressions and snapshots-as-symbols

In Niemi and Koskenniemi (2009), temporal span and granularity depend on the calendar expression under consideration. For example, *January and March 2008* is interpreted relative to the 12 months of 2008, represented as the string

(2) [y y2008 [m Jan]m [m Feb]m [m Mar]m … [m Dec]m]y

of length 39, with 4 bracket symbols]y, [y, [m and]m and the 13 symbols y2008, Jan, Feb, …, Dec. The phrase *January and March 2008* marks out the months of January and March in (2), leading to

(3) [y y2008 $\{i3$ $\{i1$ [m Jan]m $\}i1$ $\}i3$ [m Feb]m $\{i3$ $\{i2$ [m Mar]m $\}i2$ $\}i3$ …
[m Dec]m]y

with six additional symbols $\{i3$, $\{i2$, $\{i1$, $\}i3$, $\}i2$, $\}i1$. We can rewrite (3) as the string

(4) $\boxed{\text{y2008,Jan,}\dagger_3,\dagger_1}\boxed{\text{y2008,Feb}}\boxed{\text{y2008,Mar,}\dagger_3,\dagger_2}\boxed{\text{y2008,Apr}}\cdots\boxed{\text{y2008,Dec}}$

of length 12 (each box, a symbol) if we rewrite (2) as the string

(5) $\boxed{\text{y2008,Jan}}\boxed{\text{y2008,Feb}}\boxed{\text{y2008,Mar}}\cdots\boxed{\text{y2008,Dec}}$

also of length 12. In (4) and (5), a symbol is a subset of some set Φ of *fluents* (AI-speak for temporal propositions; McCarthy and Hayes 1969), enclosed by a box (rather than by the more usual curly braces $\{,\}$) to suggest a snapshot (or frame in a cartoon strip). As noted in Karttunen (2005), we can unpack such boxes, turning (5), for example, into the string

(6) [y2008 Jan] [y2008 Feb] ⋯ [y2008 Dec]

of length 48. If Φ is finite and includes neither [nor], it is trivial to devise a finite-state transducer mapping strings over the alphabet $Pow(\Phi)$ of subsets of Φ to strings over $\Phi \cup \{[,]\}$, allowing us to translate regular languages and relations freely between the alphabets $Pow(\Phi)$ and $\Phi \cup \{[,]\}$. Translating strings in Niemi and Koskenniemi (2009) to strings over $Pow(\Phi)$ is a more complicated affair and is another day's work. For now, we abstract away from implementation issues for which the strings of Niemi and Koskenniemi (2009) were carefully engineered, and work with strings of snapshots.

Strings of snapshots are arguably easier to read because snapshots are ordered chronologically (as in a cartoon or film strip). Indeed, finite subsets of $\mathbb{Z} \times P$ for LTL translate straightforwardly into strings over $Pow(P \cup \{0\})$, with, for example,

$\{(-1,p),(-1,q),(1,p),(3,p)\}$ translating to $\boxed{p,q}\boxed{0}\boxed{p}\boxed{\ }\boxed{p}$

(suggesting we equate Φ with $P \cup \{0\}$). That said, we should be careful *not* to fix the notion of time (say, to $\mathbb{Z}$) if we are to accommodate different granularities. That is,

the brackets [and] in (6) should be understood as having indeterminate granularity (in contrast to [m and]m in (2) and (3)). Moreover, we may compress (4) to

(7) $\boxed{\text{y2008,Jan,}\dagger_3}\boxed{\text{y2008,Mar,}\dagger_3}$

just as Niemi and Koskenniemi compress (3) to

$$[\text{y y2008 } \{i3\ [\text{m Jan }]\text{m }\}i3\ \{i3\ [\text{m Mar }]\text{m }\}i3\ ]\text{y}$$

with unmarked calendar periods removed (Niemi and Koskenniemi 2009: 129). Unlike the timelines depicted by (4) and (5), there are gaps between the boxes in (7) that are familiar to conceptions of time based on discrete transitions, be they the input/output pairs of programs in dynamic logic (Harel, Kozen and Tiuryn 2000) or strings of length ≥ 3 (with intermediate states) for temporal logic. In any case, it is of interest to isolate (7) from (4), or even better, (7)'s unmarked variant

(8) $\boxed{\text{y2008,Jan}}\boxed{\text{y2008,Mar}}$

from (4)'s

(9) $\boxed{\text{y2008,Jan}}\boxed{\text{y2008,Feb}}\boxed{\text{y2008,Mar}}\cdots\boxed{\text{y2008,Dec}}$

inasmuch as the mapping (9) $\mapsto$ (8) constitutes, in the words of Niemi and Koskenniemi (2009), a 'disambiguated intensional meaning' of *January and March 2008*.

1.3 Carnap–Montague intensions and partiality

The (Carnap–Montague) *intension* of an expression is a function from *indices* to *extensions* (or denotations). For the expression *January and March 2008* above, we can view the string (9) as an index, and (8) as an extension. Since Frege, it has been customary to identify the extension of a sentence (or statement) φ with one of two truth values, say True and False, essentially reducing the intension $\mathscr{I}[\varphi]$ of φ to the set of indices that $\mathscr{I}[\varphi]$ maps to True. For LTL, an index is a pair V, t where $V \subseteq \mathbb{Z} \times P$ and $t \in \mathbb{Z}$, and

$$\mathscr{I}[\varphi](V,t) = \text{True} \quad \stackrel{\text{def}}{\Longleftrightarrow} \quad V, t \models \varphi.$$

Now, the main idea behind the present work is to reconceptualize

(i) an index as a string over the alphabet $Pow(\Phi)$ built from a set Φ of fluents including $now \in \Phi$ through which we can encode, for instance, the pair

$$\{(-1,p),(-1,q),(1,p),(4,r)\},\ 2$$

as the string

$$\boxed{p,q}\boxed{0}\boxed{p}\boxed{now}\boxed{\ }\boxed{r}$$

and

(ii) an extension of a sentence φ as a set of strings over $Pow(\Phi)$, rather than True or False

Table 1. *Basic notions from an intension function* $\mathscr{I}[\varphi]$.

$\mathscr{R}[\varphi] = \{(\mathsf{s}, \mathsf{s}') \mid \mathsf{s}' \in \mathscr{I}[\varphi](\mathsf{s})\}$	Description
$\mathscr{T}[\varphi] = \text{domain}(\mathscr{R}[\varphi])$	Truth (index for entailment)
$\mathscr{A}[\varphi] = \text{image}(\mathscr{R}[\varphi])$	Aboutness (denotation as subject matter)

so that the intension $\mathscr{I}[\varphi]$ of φ maps a string $\mathsf{s} \in \mathit{Pow}(\Phi)^*$ to a set $\mathscr{I}[\varphi](\mathsf{s}) \subseteq \mathit{Pow}(\Phi)^*$ of strings that *make* φ *true at* s inasmuch as

(10) $$\varphi \text{ is true at } \mathsf{s} \overset{\text{def}}{\iff} \mathscr{I}[\varphi](\mathsf{s}) \neq \emptyset.$$

If we relax the requirement that an intension be a function and require simply that it be a binary relation between indices and extensions, then we can formulate indices and extensions alike as strings over $\mathit{Pow}(\Phi)$, identifying the intension of φ with the binary relation

$$\mathscr{R}[\varphi] \overset{\text{def}}{=} \{(\mathsf{s}, \mathsf{s}') \in \text{domain}(\mathscr{I}[\varphi]) \times \mathit{Pow}(\Phi)^* \mid \mathsf{s}' \in \mathscr{I}[\varphi](\mathsf{s})\}$$

on $\mathit{Pow}(\Phi)^*$. A string in $\mathit{Pow}(\Phi)^*$ can be understood as a situation in the sense of Barwise and Perry (1983) (as opposed to that in McCarthy and Hayes 1969). In case $\mathscr{I}[\varphi](\mathsf{s})$ consists of exactly one situation s', we can say s' is *the situation described by* φ *at* s, linking the *event time* in Reichenbach (1947) to the temporal span of s' (in contrast to the time *now* marks out in the index s). Indeed, events in Davidsonian semantics exemplify situations as truthmakers (Davidson 1967; Mulligan, Simons and Smith 1984). The domain of $\mathscr{R}[\varphi]$ consists, under (10), of the indices at which φ is true, qualifying as the *truth set* $\mathscr{T}[\varphi]$ *of* φ

$$\mathscr{T}[\varphi] \overset{\text{def}}{=} \{\mathsf{s} \in \mathit{Pow}(\Phi)^* \mid (\exists \mathsf{s}')\ \mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}'\}.$$

The image of $\mathscr{R}[\varphi]$ consists of situations that φ is *about*, making it the *about set of* φ, which we write $\mathscr{A}[\varphi]$

$$\mathscr{A}[\varphi] \overset{\text{def}}{=} \{\mathsf{s}' \in \mathit{Pow}(\Phi)^* \mid (\exists \mathsf{s})\ \mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}'\}$$

(see Table 1, Fernando 2009b).

Setting aside the problem that aboutness need not be grounded in truth, let us explore the intuition that

(11) $$\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}' \quad \text{can be read:} \quad \varphi \text{ is about } \mathsf{s}' \text{ in } \mathsf{s}.$$

It is certainly plausible to assume that $\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}'$ implies $\mathsf{s}' \sqsubseteq \mathsf{s}$, for some 'in' relation $\sqsubseteq$ on strings over $\mathit{Pow}(\Phi)$. One way to formalize (11) then is to combine the about set $\mathscr{A}[\varphi]$ with $\sqsubseteq$ for the biconditional

(12) $$\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}' \iff \mathsf{s}' \in \mathscr{A}[\varphi] \text{ and } \mathsf{s}' \sqsubseteq \mathsf{s}.$$

An immediate consequence of (12) is that for transitive relations $\sqsubseteq$, truth is *persistent* in that $\mathscr{T}[\varphi]$ is closed under $\sqsubseteq$

(13) $$\mathsf{s}_1 \in \mathscr{T}[\varphi] \text{ and } \mathsf{s}_1 \sqsubseteq \mathsf{s}_2 \quad \text{implies} \quad \mathsf{s}_2 \in \mathscr{T}[\varphi]$$

for all $\mathsf{s}_1, \mathsf{s}_2 \in Pow(\Phi)^*$. While (13) may hold for many sentences φ, there are well-known counter-examples to it, including the LTL-formula $\mathsf{G}p$ in Section 1.1. Recall that the biconditional

$$V, t \models \varphi \iff (\exists s \in \mathbf{S}[\varphi, t])\ s \subseteq V \tag{14}$$

becomes problematic when we work with bounded temporal spans short of $\mathbb{Z}$ (replacing, as it were, V with its finite subsets). To link this discussion up with (12), some notation is helpful. Given a finite subset s of $\mathbb{Z} \times P$ and an integer t, let us write $\mathsf{str}(s, t)$ for the shortest string over $Pow(P \cup \{0, now\})$ encoding the pair s, t. For instance,

$$\mathsf{str}(\{(-1, p), (-1, q), (1, p), (4, r)\},\ 2) = \boxed{p, q}\boxed{0}\boxed{p}\boxed{now}\boxed{\ }\boxed{\ }\boxed{r}.$$

The notions of 'in' $\sqsubseteq$ then becomes essentially set inclusion $\subseteq$

$$\mathsf{str}(s, t) \sqsubseteq \mathsf{str}(s', t') \iff t = t' \text{ and } s \subseteq s'$$

while 'about' $\mathscr{A}[\varphi]$ corresponds to $\bigcup_{t \in \mathbb{Z}} \mathbf{S}[\varphi, t]$

$$\mathscr{A}[\varphi] = \{\mathsf{str}(s, t) \mid t \in \mathbb{Z} \text{ and } s \in \mathbf{S}[\varphi, t]\}.$$

Now the difficulty (14) poses for $\mathsf{G}p$ when V is replaced by its finite fragments s' suggests that the best we can salvage from (14) is to step from a set $\mathbf{S}[\varphi, t]$ of relations between $\mathbb{Z}$ and P to some suitable binary relation $\mathbf{R}[\varphi, t]$ between such relations, and weaken (14) to

$$s', t \models \varphi \iff (\exists s)\ s'\ \mathbf{R}[\varphi, t]\ s. \tag{15}$$

That is, if

$$\mathscr{R}[\varphi] = \{(\mathsf{str}(s, t), \mathsf{str}(s', t)) \mid t \in \mathbb{Z} \text{ and } s\ \mathbf{R}[\varphi, t]\ s'\}$$

then (15) equates the truth set $\mathscr{T}[\varphi]$ with the domain of $\mathscr{R}[\varphi]$. But if we are to read $\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}'$ as

$$\varphi \text{ is about } \mathsf{s}' \text{ in } \mathsf{s} \tag{16}$$

then we had better beware that

$$\mathsf{s}' \in \mathscr{A}[\varphi] \quad \text{and} \quad \mathsf{s}' \sqsubseteq \mathsf{s} \tag{17}$$

falls short of (16) for sentences φ such as $\mathsf{G}p$. Picking out the sentences where (16) reduces to (17), let us define φ to be *absolute* if

$$\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}' \iff \mathsf{s}' \in \mathscr{A}[\varphi] \text{ and } \mathsf{s}' \sqsubseteq \mathsf{s} \tag{18}$$

for all $\mathsf{s}, \mathsf{s}' \in Pow(\Phi)^*$. The LTL-formulas p and $previous(q)$ are absolute; $\mathsf{G}p$ is not.

Whether or not φ is absolute, it is useful for $\mathscr{R}[\varphi]$ to be a regular relation because it gives us a computational handle not only on its image, the about set $\mathscr{A}[\varphi]$, but also on its domain, the truth set $\mathscr{T}[\varphi]$. Entailment $\vdash$ between sentences is often relativized to constraints C, which we can identify with a set of strings (following,

for example, Beesley and Karttunen 2003) to define

$$\varphi \vdash_C \psi \overset{\text{def}}{\iff} (\forall \mathsf{s} \in C)\ \varphi \text{ is true at } \mathsf{s} \text{ implies } \psi \text{ is true at } \mathsf{s}$$
$$\iff C \cap \mathscr{T}[\varphi] \ \subseteq\ \mathscr{T}[\psi].$$

If $\mathscr{T}[\varphi], \mathscr{T}[\psi]$ and C are all regular languages, then $\vdash_C$ above is decidable (since inclusion between regular languages is). Note that the strings in $\mathscr{T}[\varphi]$ are finite, whereas the relations $V \subseteq \mathbb{Z} \times P$ in LTL-semantics can be infinite, collectively inducing the entailments

$$\varphi \vdash^{\text{LTL}} \psi \overset{\text{def}}{\iff} (\forall V \subseteq \mathbb{Z} \times P)(\forall t \in \mathbb{Z})\ V, t \models \varphi \text{ implies } V, t \models \psi.$$

1.4 Claims and some finite-state matters

The bounded entailments $\vdash_C$ provide approximations of $\vdash^{\text{LTL}}$ that we take up in Section 3. But far from providing a new, improved way of calculating $\vdash^{\text{LTL}}$ (which we do not), the point of introducing relations $\mathscr{R}[\varphi]$ is to reason *beyond* $\vdash^{\text{LTL}}$,

(a1) breaking free from the tyranny of the clock $\mathbb{Z}$ built into the semantics behind $\vdash^{\text{LTL}}$

and

(a2) bringing out the about sets $\mathscr{A}[\varphi]$ that are at best implicit in that semantics.

The 'tyranny of the clock $\mathbb{Z}$' mentioned in (a1) is twofold: its span is infinite (two ways), and its ticks (grain) are fixed by the successor (plus-1) relation (in $\mathbb{Z}$). The focus below is on bounding span, although a few words about grain are offered at the conclusion on which I hope to expand elsewhere.

As for (a2), the proposed relational semantics $\mathscr{R}[\varphi]$ consists not only of the indices in $\mathscr{T}[\varphi]$ that shape $\varphi \vdash_C \psi$, but also of denotations in $\mathscr{A}[\varphi]$, exemplified by events (e.g. Fernando 2009b). It is notable that Hans Kamp, famous in temporal logic for *since* and *until*, leaves temporal logic out of *Discourse Representation Theory* (Kamp and Reyle 1993: 492) and works instead with events. As useful as it has proved in computer science, LTL is an artificial language, unfit for linguistic analyses that posit, for instance, two forms of *until*, dependent on the (a)telicity of event descriptions, featuring presuppositional as well as assertional dimensions (e.g. Karttunen 1974; Condoravdi 2009). Introducing relations $\mathscr{R}[\varphi]$ moves us, I claim, closer to an account of events (as denotations) and presuppositions (via indices). This is not a small claim, and I shall confine myself here to events as denotations. Analyzing presuppositions as requirements on indices calls for (in my view) adjustments to the definitions of $\mathscr{R}[\varphi]$ given below.

Presuppositions and telicity aside, let us consider the past tense, and ask if the LTL clause

$$V, t \models \mathsf{P}\varphi \overset{\text{def}}{\iff} (\exists t' \leq t)\ V, t' \models \varphi$$

supports a reading of $\mathsf{P}\varphi$ as 'φ in the present or in the past'? It is plausible enough in case φ is an atomic formula p for which $V(t, p)$ says there is a V-situation of type

Table 2. *Some regular notions given a regular language L and relation R.*

$\mathsf{has\text{-}factor} = \{(\mathsf{s},\mathsf{s}') \mid (\exists \mathsf{u},\mathsf{v})\ \mathsf{s} = \mathsf{us}'\mathsf{v}\}$	Factor
$R_L = \{(\mathsf{s},\mathsf{s}') \in R \mid \mathsf{s}' \in L\}$	L-restriction of R
$\langle R\rangle L = \mathrm{domain}(R_L)$	Peirce product
$\overline{L} = Pow(\Phi)^* - L$	Complement of L
$[R]L = \overline{\langle R\rangle \overline{L}}$	Dual of Peirce product

p with time t. But suppose φ were constructed from the connective *until*, under the LTL clause

$$V,t \models \psi \ \textit{until}\ \chi \overset{\text{def}}{\iff} (\exists t' \geq t)\ V,t' \models \chi \text{ and whenever } t \leq t'' < t',\ V,t'' \models \psi$$

(such as it is[2]). It is natural to assume that part of a V-situation of type ψ *until* p is a V-situation of type p. But for $V_\circ = \{(0,q),(1,q),(2,q),(3,p)\}$, we have

$$V_\circ, 1 \models \mathsf{P}(q\ \textit{until}\ p)$$

even though every $V_\circ$-situation of type q *until* p stretches to 3, which is not in the past of 1. In general, for non-atomic φ, one cannot conclude from $V,t' \models \varphi$ and $t' < t$ that there is a V-situation of type φ in the past of t. To report φ in the past tense at speech time t in V, what we require is *not* some $t' < t$ such that $V,t' \models \varphi$ but a truthmaker for φ in the past of t.[3] That is, it is not so much the domain $\mathscr{T}[\varphi]$ of $\mathscr{R}[\varphi]$ that is relevant, but rather the image $\mathscr{A}[\varphi]$ of $\mathscr{R}[\varphi]$ and the 'in' relation $\sqsubseteq$.

The next section fleshes out relations $\mathscr{R}[\varphi]$ and $\sqsubseteq$ in finite-state terms. Under the conventions adopted in, for example, *Regular Model Checking* (Bouajjani *et al.* 2000), regular relations can hold only between strings of the same length. But for $\sqsubseteq$ (for instance) to be regular, we drop this requirement, and follow Beesley and Karttunen (2003) in permitting a finite-state transducer to make componentwise ϵ-moves (where ϵ is the null/empty string). A notable difference between Bouajjani *et al.* (2000) and Beesley and Karttunen (2003) is that regular relations are closed under intersection in the former but not in the latter. What is important for our purposes, however, is that the basic notions summarized in Table 2 are regular. A *factor of* s is a string s' such that $\mathsf{s} = \mathsf{us}'\mathsf{v}$ for some strings u and v. The relation $\mathsf{has\text{-}factor}$ defined by

$$\mathsf{s}\ \mathsf{has\text{-}factor}\ \mathsf{s}' \overset{\text{def}}{\iff} (\exists \mathsf{u},\mathsf{v})\ \mathsf{s} = \mathsf{us}'\mathsf{v}$$

is computed by the finite-state transducer with three states $0,1,2$, all final (accepting), 0 initial, with loops

$$0 \xrightarrow{\alpha:\epsilon} 0 \quad \text{and} \quad 2 \xrightarrow{\alpha:\epsilon} 2 \quad \text{for every symbol } \alpha$$

[2] The shortcomings of this semantics as an account of English *until* are irrelevant to the present point; the reader is free to rename the connective characterized by the clause.

[3] A Reichenbachian account of tense and aspect that is free from this Priorean defect is formulated in terms of strings in section 2 of Fernando (2008a).

plus transitions

(19) $$0 \overset{\alpha:\alpha}{\to} 1 \overset{\alpha:\alpha}{\to} 1 \overset{\alpha:\alpha}{\to} 2 \quad \text{for every symbol } \alpha.$$

Given a finite-state transducer for R with transitions $\to_R$ and a finite automaton for L with transitions $\to_L$, we can form a finite-state transducer for the (*right*) *L-restriction* R_L *of* R defined by

$$R_L \overset{\text{def}}{=} \{(\mathsf{s}, \mathsf{s}') \in R \mid \mathsf{s}' \in L\}$$

by collecting the transitions

(20) $$(q, q') \overset{\alpha:\alpha'}{\to} (r, r') \quad \text{for} \quad q \overset{\alpha:\alpha'}{\to}_R r \text{ and } (q' \overset{\alpha'}{\to}_L r' \text{ or } (\alpha' = \epsilon \text{ and } q' = r'))$$

(as in the usual construction for the intersection of regular languages, but with ϵ-moves). The domain of R_L is the *Peirce product* $\langle R \rangle L$ *of* R *with* L[4]

$$\langle R \rangle L \overset{\text{def}}{=} \{\mathsf{s} \mid (\exists \mathsf{s}')\ \mathsf{s}\ R_L\ \mathsf{s}'\},$$

for which we existentially quantify α' out from (20) for

$$(q, q') \overset{\alpha}{\to} (r, r') \quad \text{whenever} \quad (\exists \alpha')\ q \overset{\alpha:\alpha'}{\to}_R r \text{ and } (q' \overset{\alpha'}{\to}_L r' \text{ or } (\alpha' = \epsilon \text{ and } q' = r')).$$

And as regular languages are closed under Boolean operations, including complementation $\overline{L} \overset{\text{def}}{=} \Sigma^* - L$ over the alphabet Σ, we can form the dual $[R]L$ of the Peirce product $\langle R \rangle L$

$$\begin{aligned} [R]L &\overset{\text{def}}{=} \overline{\langle R \rangle \overline{L}} \\ &= \{\mathsf{s} \in \Sigma^* \mid (\forall \mathsf{s}' \text{ such that } \mathsf{s}\ R\ \mathsf{s}')\ \mathsf{s}' \in L\} \end{aligned}$$

just as $\forall$ is the dual of $\exists$.

In the remainder of this paper, strings are formed from the alphabet $\Sigma = Pow(\Phi)$, under the assumption that Φ is finite, so that there are only finitely many subsets of Φ (i.e. symbols) which we write α, α', β, etc. That is, a string $\mathsf{s} \in Pow(\Phi)^*$ is a sequence of the form $\alpha_1 \cdots \alpha_n$, while a symbol α is a box of fluents (from Φ). Largely a matter of convenience, this choice of alphabet provides a level of abstraction that simplifies many of the constructions below.

2 Relations $\mathscr{R}[\varphi]$ for some absolute formulas φ

This section starts with a simple generalization of set inclusion to strings over the alphabet $Pow(\Phi)$, which is then used to formulate constraints such as

$$\boxed{\varphi \wedge \psi} \Rightarrow \boxed{\varphi, \psi}$$

picking out strings that contain $\varphi \wedge \psi$ only if they contain φ and ψ in the same box. Some refinements are then made to capture the 'in' relation $\sqsubseteq$ and define $\mathscr{R}[\varphi]$ for various absolute sentences φ.

[4] The terminology 'Peirce product' is from Brink, Britz, and Schmidt (1994), but the notation $\langle R \rangle L$ is borrowed from dynamic logic (Harel *et al.* 2000).

2.1 Subsumption and constraints

Checking set inclusion $\supseteq$ componentwise between strings from $Pow(\Phi)^*$ of the same length, let

$$\alpha_1 \cdots \alpha_n \ \trianglerighteq\ \alpha'_1 \cdots \alpha'_m \overset{\text{def}}{\iff} n = m \text{ and } \alpha_i \supseteq \alpha'_i \text{ for } 1 \leq i \leq n$$

for all symbols $\alpha_i, \alpha'_i \subseteq \Phi$. (Exactly what is in Φ does not matter just now, as long as Φ is finite.) In Fernando (2004), $\mathsf{s} \trianglerighteq \mathsf{s}'$ is pronounced: s subsumes s'. The relation is computed by the finite-state transducer with a single state 0, both initial and final, and transitions

$$0 \xrightarrow{\alpha:\alpha'} 0 \quad \text{for} \quad \alpha' \subseteq \alpha \subseteq \Phi.$$

Next, given a language L over $Pow(\Phi)$, we let $\mathsf{s} \trianglerighteq L$ mean s belongs to the Peirce product $\langle\trianglerighteq\rangle L$ (read: s subsumes L)

$$\begin{aligned} \mathsf{s} \trianglerighteq L &\overset{\text{def}}{\iff} \mathsf{s} \in \langle\trianglerighteq\rangle L \\ &\iff (\exists \mathsf{s}' \in L)\ \mathsf{s} \trianglerighteq \mathsf{s}'. \end{aligned}$$

Then, given another language L' over $Pow(\Phi)$, let $L \Rightarrow L'$ be the set of strings s such that every factor of s that subsumes L also subsumes L'

$$L \Rightarrow L' \overset{\text{def}}{=} \{\mathsf{s} \in Pow(\Phi)^* \mid \text{for every factor } \mathsf{s}' \text{ of } \mathsf{s}, \mathsf{s}' \trianglerighteq L \text{ implies } \mathsf{s}' \trianglerighteq L'\}.$$

Collecting the factors of s that subsume L in the $\langle\trianglerighteq\rangle L$-restriction of has-factor

$$\begin{aligned} \mathsf{s} \text{ has-factor}_{\langle\trianglerighteq\rangle L}\ \mathsf{s}' &\overset{\text{def}}{\iff} \mathsf{s} \text{ has-factor } \mathsf{s}' \text{ and } \mathsf{s}' \in \langle\trianglerighteq\rangle L \\ &\iff \mathsf{s} \text{ has-factor } \mathsf{s}' \text{ and } \mathsf{s}' \trianglerighteq L, \end{aligned}$$

it follows that

$$L \Rightarrow L' = [\text{has-factor}_{\langle\trianglerighteq\rangle L}]\langle\trianglerighteq\rangle L'.$$

From the regularity of the constructs in Table 2, we can conclude that $L \Rightarrow L'$ is a regular language if L and L' are. Indeed,

$$L \Rightarrow L' = \overline{Pow(\Phi)^*\ (\langle\trianglerighteq\rangle L \cap \overline{\langle\trianglerighteq\rangle L'})\ Pow(\Phi)^*}.$$

Among the constraints we can formulate with $\Rightarrow$ are

$$\boxed{\varphi \vee \psi} \Rightarrow \boxed{\varphi} \mid \boxed{\psi}$$

picking out strings that contain $\varphi \vee \psi$ only if they contain φ or ψ in the same box,

$$\boxed{\ } \Rightarrow \boxed{\varphi} \mid \boxed{\overline{\varphi}}$$

requiring either φ or its negation $\overline{\varphi}$ to be in every box. and

$$\boxed{\varphi, \overline{\varphi}} \Rightarrow \emptyset$$

banning φ and $\overline{\varphi}$ from occurring in the same box.

We can also form

$$\boxed{next(\varphi)}\boxed{\ } \Rightarrow \boxed{\ }\boxed{\varphi}$$

to put φ at every box succeeding one containing $next(\varphi)$, in line with the LTL clause

$$V, t \models next(\varphi) \stackrel{\text{def}}{\iff} V, t+1 \models \varphi.$$

But to require that there *be* a box after every box containing $next(\varphi)$, we form

$$\boxed{next(\varphi)} \stackrel{\mathsf{a}}{\Rightarrow} \boxed{\varphi},$$

where $L \stackrel{\mathsf{a}}{\Rightarrow} L'$ is pronounced 'L' after every L' and

$$L \stackrel{\mathsf{a}}{\Rightarrow} L' \stackrel{\text{def}}{=} \{\mathsf{s} \in Pow(\Phi)^* \mid \text{after every factor of } s \text{ that subsumes } L \text{ is a string that subsumes } L'\}.$$

Defining

$$\mathsf{s}\ after^L\ \mathsf{s}' \stackrel{\text{def}}{\iff} (\exists \mathsf{u} \trianglerighteq \Box^* L)\ \mathsf{s} = \mathsf{us}',$$

we get

$$L \stackrel{\mathsf{a}}{\Rightarrow} L' = [after^L]\ \langle\trianglerighteq\rangle(L'\Box^*).$$

It is not difficult to convert a finite automaton for L into a finite-state transducer for $after_L$. Hence, $L \stackrel{\mathsf{a}}{\Rightarrow} L'$ is regular if L and L' are. In fact,

$$L \stackrel{\mathsf{a}}{\Rightarrow} L' = \overline{\langle\trianglerighteq\rangle(\Box^* L)}\ \overline{\langle\trianglerighteq\rangle(L'\Box^*)}.$$

Modulo subsumption $\trianglerighteq$, $L \stackrel{\mathsf{a}}{\Rightarrow} L'$ is one form of Koskenniemi's restrictions (Beesley and Karttunen 2003), a second one being $L \stackrel{\mathsf{b}}{\Rightarrow} L'$, read '$L'$ before every L,' defined by

$$L \stackrel{\mathsf{b}}{\Rightarrow} L' \stackrel{\text{def}}{=} [before^L]\ \langle\trianglerighteq\rangle(\Box^* L'),$$

where

$$\mathsf{s}\ before^L\ \mathsf{s}' \stackrel{\text{def}}{\iff} (\exists \mathsf{v} \trianglerighteq L\Box^*)\ \mathsf{s} = \mathsf{s}'\mathsf{v}.$$

As with $\Rightarrow$ and $\stackrel{\mathsf{a}}{\Rightarrow}$, $L \stackrel{\mathsf{b}}{\Rightarrow} L'$ is regular if L and L' are, with

$$L \stackrel{\mathsf{b}}{\Rightarrow} L' = \overline{\langle\trianglerighteq\rangle(\Box^* L')}\ \overline{\langle\trianglerighteq\rangle(L\Box^*)}.$$

We can also strengthen $\boxed{\ \boxed{previous(\varphi)}\ } \Rightarrow \boxed{\ \varphi\ }$ to

$$\boxed{previous(\varphi)} \stackrel{\mathsf{b}}{\Rightarrow} \boxed{\varphi}$$

for the LTL clause

$$V, t \models previous(\varphi) \stackrel{\text{def}}{\iff} V, t-1 \models \varphi.$$

Both $\Rightarrow$ and $\stackrel{\mathsf{a}}{\Rightarrow}$ are useful to analyze $\varphi\ until\ \psi$ through auxiliary formulas $\varphi\ ntil\ \psi$

$$\boxed{\varphi\ until\ \psi} \Rightarrow \boxed{\psi} \mid \boxed{\varphi,\ \varphi\ ntil\ \psi}$$

with

$$\boxed{\varphi\ ntil\ \psi} \stackrel{\mathsf{a}}{\Rightarrow} \boxed{\varphi}^*\boxed{\psi}.$$

(Thus, if p and q are atomic formulas, we can picture p *until* q roughly as $\boxed{p}^*\boxed{q}$.) Similarly, for *since* under the LTL clause

$$V,t \models \varphi \textit{ since } \psi \overset{\text{def}}{\iff} (\exists t' \leq t)(V,t' \models \psi \text{ and whenever } t' < t'' \leq t,\ V,t'' \models \varphi)$$

we apply $\overset{\mathsf{b}}{\Rightarrow}$ and auxiliary formulas φ *sinc* ψ

$$\boxed{\varphi \textit{ since } \psi} \Rightarrow \boxed{\psi} \mid \boxed{\varphi,\ \varphi \textit{ sinc } \psi}$$

$$\boxed{\varphi \textit{ sinc } \psi} \overset{\mathsf{b}}{\Rightarrow} \boxed{\psi}\boxed{\varphi}^*.$$

If we fix a tautology $\top$ with the vacuous constraint

$$\boxed{\top} \Rightarrow \boxed{\ }$$

then the Priorean past $\mathsf{P}\varphi$ can be understood as an abbreviation of $\top$ *since* φ, and its future counterpart $\mathsf{F}\varphi$ as $\top$ *until* φ.

2.2 Unpadding for information content

Although the relation $\trianglerighteq$ is just right for formulating constraints, it is convenient to weaken it slightly to compare strings of different lengths. A candidate for the converse of the 'in' relation $\sqsubseteq$ from Section 1.3 satisfying

$$\mathsf{str}(s,t) \sqsupseteq \mathsf{str}(s',t') \iff t = t' \text{ and } s \supseteq s'$$

for all finite $s,s' \subseteq \mathbb{Z} \times P$ and $t,t' \in \mathbb{Z}$ is the relation $\sqsupseteq_\circ$ composing has-factor with $\trianglerighteq$

$$\begin{aligned} \mathsf{s} \sqsupseteq_\circ \mathsf{s}' &\overset{\text{def}}{\iff} \mathsf{s} \text{ has-factor}; \trianglerighteq \mathsf{s}' \\ &\iff (\exists \mathsf{u},\mathsf{s}'',\mathsf{v})\ \mathsf{s} = \mathsf{us}''\mathsf{v} \text{ and } \mathsf{s}'' \trianglerighteq \mathsf{s}'. \end{aligned}$$

A finite-state transducer for $\sqsupseteq_\circ$ can be built from the one described in Section 1.4 above for has-factor by generalizing the transitions in line (19) to

$$0 \overset{\alpha:\alpha'}{\to} 1 \overset{\alpha:\alpha'}{\to} 1 \overset{\alpha:\alpha'}{\to} 2 \quad \text{for } \alpha' \subseteq \alpha \subseteq \Phi.$$

The relation $\sqsupseteq_\circ$ has the defect, however, that $\mathsf{s} \sqsupseteq \mathsf{s}'$ may fail to hold because of empty boxes in s' — e.g. $\boxed{\textit{now}} \not\sqsupseteq_\circ \boxed{\ }\boxed{\textit{now}}$. Accordingly, we unpad a string $\mathsf{s} \in \mathit{Pow}(\Phi)^*$ by deleting all initial and final $\Box$'s

$$\mathit{unpad}(\mathsf{s}) \overset{\text{def}}{=} \begin{cases} \mathsf{s} & \text{if } \mathsf{s} \text{ neither begins nor ends with } \Box \\ \mathit{unpad}(\mathsf{s}') & \text{if } \mathsf{s} = \Box\mathsf{s}' \text{ or else if } \mathsf{s} = \mathsf{s}'\Box \end{cases}$$

so that for example, $\mathit{unpad}(\boxed{\ }\boxed{p}\boxed{\ }\boxed{\textit{now}}\boxed{\ }\boxed{\ }) = \boxed{p}\boxed{\ }\boxed{\textit{now}}$. Next, we say s and s' are *unpad-equivalent* if their unpadding is identical

$$\mathsf{s} \approx \mathsf{s}' \overset{\text{def}}{\iff} \mathit{unpad}(\mathsf{s}) = \mathit{unpad}(\mathsf{s}')$$

and say a string s *weakly subsumes* s', written $\mathsf{s} \sqsupseteq \mathsf{s}'$, if some string *unpad*-equivalent to s subsumes some string *unpad*-equivalent to s'

$$\mathsf{s} \sqsupseteq \mathsf{s}' \overset{\text{def}}{\iff} (\exists \mathsf{s}_\circ \approx \mathsf{s})(\exists \mathsf{s}'_\circ \approx \mathsf{s}')\ \mathsf{s}_\circ \trianglerighteq \mathsf{s}'_\circ.$$

Table 3. *Comparing information content*

unpad-equivalent	$\mathsf{s} \approx \mathsf{s}' \iff \mathit{unpad}(\mathsf{s}) = \mathit{unpad}(\mathsf{s}')$
Subsume	$\alpha_1 \cdots \alpha_n \trianglerighteq \alpha'_1 \cdots \alpha'_m \iff n = m$ and $\alpha_i \sqsupseteq \alpha'_i$ for $1 \leq i \leq n$
Factor-subsume	$\mathsf{s} \sqsupseteq_\circ \mathsf{s}' \iff \mathsf{s}$ has-factor$; \trianglerighteq \mathsf{s}'$
Weakly subsume	$\mathsf{s} \sqsupseteq \mathsf{s}' \iff (\exists \mathsf{s}_\circ \approx \mathsf{s})(\exists \mathsf{s}'_\circ \approx \mathsf{s}')\ \mathsf{s}_\circ \trianglerighteq \mathsf{s}'_\circ$
	$\iff \mathsf{s} \sqsupseteq_\circ \mathit{unpad}(\mathsf{s}')$

The relations *unpad* and $\approx$ are regular, and as

$$\begin{aligned} \mathsf{s} \sqsupseteq \mathsf{s}' &\iff (\exists \mathsf{s}'_\circ \approx \mathsf{s}')\ \mathsf{s} \trianglerighteq \mathsf{s}'_\circ \\ &\iff \mathsf{s} \sqsupseteq_\circ \mathit{unpad}(\mathsf{s}') \end{aligned}$$

we can turn the finite-state transducer for $\sqsupseteq_\circ$ above into one for $\sqsupseteq$ by adding the loops

$$0 \xrightarrow{\epsilon:\square} 0 \quad \text{and} \quad 2 \xrightarrow{\epsilon:\square} 2.$$

Of course, if we arrange that $\mathsf{s}' = \mathit{unpad}(\mathsf{s}')$ for strings s' that we put to the right of $\sqsupseteq$, we can make do with $\sqsupseteq_\circ$ instead of $\sqsupseteq$.

It is natural to think of strings in a language as possibilities in the same way that worlds in a proposition are under possible worlds semantics (or models of a sentence are in model-theoretic semantics). We lift $\sqsupseteq$ to languages L, L' through the Peirce product

$$\begin{aligned} L \sqsupseteq L' &\overset{\text{def}}{\iff} L \subseteq \langle \sqsupseteq \rangle L' \\ &\iff (\forall \mathsf{s} \in L)(\exists \mathsf{s}' \in L')\ \mathsf{s} \sqsupseteq \mathsf{s}' \end{aligned}$$

paralleling the definition in possible worlds semantics that a proposition A entails a proposition A' if $A \subseteq A'$. To weed out spurious possibilities, it is useful to bring in a language C to intersect with $\langle \sqsupseteq \rangle L$ for

$$C[L] \overset{\text{def}}{=} C \cap \langle \sqsupseteq \rangle L$$

and then beef up $L \sqsupseteq L'$ to an entailment

$$\begin{aligned} L \vdash_C L' &\overset{\text{def}}{\iff} C[L] \ \sqsupseteq\ L' \\ &\iff C \cap \langle \sqsupseteq \rangle L \ \subseteq\ \langle \sqsupseteq \rangle L'. \end{aligned}$$

Clearly, $L \vdash_C L'$ whenever $L \sqsupseteq L'$. The introduction of C allows us not only to enlarge a string in L to one in $\langle \sqsupseteq \rangle L$, but also to restrict attention to strings meeting the membership conditions for C

$$L \vdash_C L' \iff (\forall \mathsf{s} \in C)\ \mathsf{s} \sqsupseteq L \text{ implies } \mathsf{s} \sqsupseteq L'.$$

These membership conditions can be viewed as constraints (to satisfy), as we see next.

2.3 Some absolute formulas

A sentence φ is absolute if its description relation $\mathscr{R}[\varphi]$ is the $\mathscr{A}[\varphi]$-restriction of $\sqsupseteq$, making its truth set $\mathscr{T}[\varphi]$ the Peirce product $\langle\sqsupseteq\rangle\mathscr{A}[\varphi]$. We now flesh out $\mathscr{A}[\varphi]$ for certain absolute LTL-formulas φ—in particular, those belonging to the set P_+ of formulas that can be formed from a set P of atomic formulas using the connectives $\wedge, \vee, \mathit{next}, \mathit{previous}, \mathit{until}, \mathit{ntil}, \mathit{since}, \mathit{sinc}$. We start with constraints $\mathscr{C}(\varphi)$ for $\varphi \in P_+$, setting

$$\begin{aligned}
\mathscr{C}(\varphi \wedge \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap \mathscr{C}(\psi) \cap (\boxed{\varphi \wedge \psi} \Rightarrow \boxed{\varphi, \psi}) \\
\mathscr{C}(\varphi \vee \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap \mathscr{C}(\psi) \cap (\boxed{\varphi \vee \psi} \Rightarrow \boxed{\varphi} \| \boxed{\psi}) \\
\mathscr{C}(\mathit{next}(\varphi)) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap (\boxed{\mathit{next}(\varphi)} \stackrel{\text{a}}{\Rightarrow} \boxed{\varphi}) \\
\mathscr{C}(\mathit{previous}(\varphi)) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap (\boxed{\mathit{previous}(\varphi)} \stackrel{\text{b}}{\Rightarrow} \boxed{\varphi}) \\
\mathscr{C}(\varphi \ \mathit{until}\ \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi \ \mathit{ntil}\ \psi) \cap (\boxed{\varphi \ \mathit{until}\ \psi} \Rightarrow \boxed{\psi} \| \boxed{\varphi,\ \varphi \ \mathit{ntil}\ \psi}) \\
\mathscr{C}(\varphi \ \mathit{ntil}\ \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap \mathscr{C}(\psi) \cap (\boxed{\varphi \ \mathit{ntil}\ \psi} \stackrel{\text{a}}{\Rightarrow} \boxed{\varphi}^* \boxed{\psi}) \\
\mathscr{C}(\varphi \ \mathit{since}\ \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi \ \mathit{sinc}\ \psi) \cap (\boxed{\varphi \ \mathit{since}\ \psi} \Rightarrow \boxed{\psi} \| \boxed{\varphi,\ \varphi \ \mathit{sinc}\ \psi}) \\
\mathscr{C}(\varphi \ \mathit{sinc}\ \psi) &\stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap \mathscr{C}(\psi) \cap (\boxed{\varphi \ \mathit{sinc}\ \psi} \stackrel{\text{b}}{\Rightarrow} \boxed{\psi} \boxed{\varphi}^*)
\end{aligned}$$

and for $p \in P$,

$$\mathscr{C}(p) \stackrel{\text{def}}{=} \boxed{\mathit{now}} \boxed{}^* \boxed{\mathit{now}} \ \Rightarrow\ \emptyset$$

saying the fluent *now* occurs at most once. Next, we intersect $\mathscr{C}(\varphi)$ with the set $\langle\sqsupseteq\rangle\boxed{\mathit{now}, \varphi}$ of strings in which φ and *now* occur in a box, to form

$$\mathscr{A}_\circ(\varphi) \stackrel{\text{def}}{=} \mathscr{C}(\varphi) \cap \langle\sqsupseteq\rangle\boxed{\mathit{now}, \varphi}.$$

But what alphabet $\mathit{Pow}(\Phi)$ are we assuming for $\mathscr{C}(\varphi)$ and $\langle\sqsupseteq\rangle\boxed{\mathit{now}, \varphi}$? Φ had better be finite if the Peirce product $\langle\sqsubset\rangle L$ is to be regular. It suffices (for our purposes) to let Φ be the finite subset $\Psi(\varphi)$ of $P_+ \cup \{\mathit{now}\}$ consisting of φ and formulas mentioned in the definition of $\mathscr{C}(\varphi)$ — roughly φ, its subformulas and *now*. For example,

$$\begin{aligned}
\Psi(\varphi \vee \psi) &= \{\varphi \vee \psi\} \cup \Psi(\varphi) \cup \Psi(\psi) \\
\Psi(\varphi \ \mathit{since}\ \psi) &= \{\varphi \ \mathit{since}\ \psi\} \cup \Psi(\varphi \ \mathit{sinc}\ \psi) \\
\Psi(\varphi \ \mathit{sinc}\ \psi) &= \{\varphi \ \mathit{sinc}\ \psi\} \cup \Psi(\varphi) \cup \Psi(\psi) \\
\Psi(p) &= \{p, \mathit{now}\}
\end{aligned}$$

so that for $p, q, r \in P$,

$$\begin{aligned}
\Psi((p \vee q) \ \mathit{since}\ \mathit{next}(r)) = \{&(p \vee q) \ \mathit{since}\ \mathit{next}(r),\ (p \vee q) \ \mathit{sinc}\ \mathit{next}(r), \\
&p \vee q,\ \mathit{next}(r),\ p, q, r, \mathit{now}\}.
\end{aligned}$$

Even with the alphabet restricted, the language $\mathscr{A}_\circ(\varphi)$ is quite massive. But we can reduce it a couple of ways. The first is through $\trianglerighteq$-minimization: given a language

L, define the set $L_{\trianglerighteq}$ of $\trianglerighteq$-*minimal strings in* L by

$$L_{\trianglerighteq} \stackrel{\text{def}}{=} L - \langle \triangleright \rangle L,$$

where $\triangleright$ is $\trianglerighteq$ minus equality

$$\mathsf{s} \triangleright \mathsf{s}' \stackrel{\text{def}}{\iff} \mathsf{s} \trianglerighteq \mathsf{s}' \text{ and } \mathsf{s} \neq \mathsf{s}'.$$

For example,

$$(\square^* | L)_{\trianglerighteq} = \square^*.$$

Notice that $L_{\trianglerighteq}$ is regular if L is (as a finite-state transducer for $\triangleright$ can be easily obtained from the one above for $\trianglerighteq$ by requiring that there be a proper inclusion $\supset$). The second way of trimming a language is by projecting every string $\alpha_1 \cdots \alpha_n$ in it to the string

$$\rho(\alpha_1 \cdots \alpha_n) \stackrel{\text{def}}{=} (\alpha_1 \cap (P \cup \{now\})) \cdots (\alpha_n \cap (P \cup \{now\}))$$

restricting the symbols to subsets of $P \cup \{now\}$. For instance, if $p \in P$ then

$$\rho(\boxed{p, \psi \vee \varphi \mid now, previous(\chi)}) = \boxed{p \mid now}.$$

Clearly, as a relation, ρ is regular. Now, for $\varphi \in P_+$, we take the $\trianglerighteq$-minimal strings in $\mathscr{A}_\circ(\varphi)$, and apply ρ and *unpad* in succession to form

$$\mathscr{A}[\varphi] \stackrel{\text{def}}{=} \{unpad(\rho(\mathsf{s})) \mid \mathsf{s} \in \mathscr{A}_\circ(\varphi)_{\trianglerighteq}\}.$$

From the closure properties of regular languages, it follows that $\mathscr{A}[\varphi]$ is regular, as is $\mathscr{R}[\varphi] = \langle \sqsubseteq \rangle \mathscr{A}[\varphi]$. To relate $\mathscr{A}[\varphi]$ to LTL, let us adopt the obvious interpretation of the connectives *ntil*

$$V, t \models \varphi \; ntil \; \psi \stackrel{\text{def}}{\iff} (\exists t' > t)\; V, t' \models \psi \text{ and whenever } t < t'' < t',\; V, t'' \models \varphi$$

and *sinc*

$$V, t \models \varphi \; sinc \; \psi \stackrel{\text{def}}{\iff} (\exists t' < t)\; V, t' \models \psi \text{ and whenever } t' < t'' < t,\; V, t'' \models \varphi$$

against relations $V \subseteq \mathbb{Z} \times P$ and integers $t \in \mathbb{Z}$. Moreover, as 0 is not one of the fluents out of which we construct strings in $\mathscr{A}[\varphi]$, we delete the fluent 0 in our encoding $\mathsf{str}(s, t)$ of a finite relation $s \subseteq \mathbb{Z} \times P$ and an integer $t \in \mathbb{Z}$ to form the string $\mathsf{str}_\circ(s, t)$. For example,

$$\begin{aligned} \mathsf{str}_\circ(\{(-1, p), (-1, q), (2, r)\}, 1) &= \boxed{p, q \mid \mid now \mid r} \\ &= \mathsf{str}_\circ(\{(-2, p), (-2, q), (1, r)\}, 0) \end{aligned}$$

illustrating the possibility that *now* can be set without loss of generality to 0.[5] To establish the faithfulness of the about sets $\mathscr{A}[\varphi]$ to LTL, we can prove by induction on $\varphi \in P_+$ that

[5] That is, for all LTL-formulas φ, relations $V \subseteq \mathbb{Z} \times P$ and integers t,

$$V, t \models \varphi \iff V_t, 0 \models \varphi$$

Theorem. *For all* $\varphi \in P_+$, $V \subseteq \mathbb{Z} \times P$ *and* $t \in \mathbb{Z}$,

$$V, t \models \varphi \iff (\exists s \subseteq V)\ \mathsf{str}_\circ(s, t) \in \mathscr{A}[\varphi].$$

2.4 Irregular replace

All the clean-up on $\mathscr{A}_\circ(\varphi)$ above just to extract $\mathscr{A}[\varphi]$ invites the question: why not define a regular *relation* unwinding a non-atomic formula φ rather than a regular *language* $\mathscr{A}[\varphi]$ via a system of constraints and projections? (Or for readers familiar with Beesley and Karttunen (2003), why not replace, rather than constrain?) Take, for example, the formula $\mathsf{P}p$ saying p holds now or sometime in the past. It is easy enough to construct a finite-state transducer for a relation $\hat{R}$ replacing $\mathsf{P}p$ by p or shifting $\mathsf{P}p$ backwards one square so that, for example,

$$\boxed{\ }\boxed{\mathsf{P}p, q}\ \hat{R}\ \boxed{\ }\boxed{p, q} \quad \text{and} \quad \boxed{\ }\boxed{\mathsf{P}p, q}\ \hat{R}\ \boxed{\mathsf{P}p}\boxed{q}.$$

Repeatedly applying $\hat{R}$, let $\mathrm{tc}(\hat{R})$ be the transitive closure of $\hat{R}$. We have for all non-negative integers m and n,

$$\boxed{\ }^{m+n+1}\boxed{\mathsf{P}p, q} \quad \mathrm{tc}(\hat{R}) \quad \boxed{\ }^{m}\boxed{p}\boxed{\ }^{n}\boxed{q}.$$

The difficulty, however, is $\mathrm{tc}(\hat{R})$ is not regular, even though $\hat{R}$ is. To see this, consider feeding strings from the language $\boxed{\ }^{*}\boxed{\mathsf{P}p, q}^{*}$ as inputs to $\mathrm{tc}(\hat{R})$. What output strings do we get that belong in the language $\boxed{p}^{+}\boxed{q}^{+}$ (where $L^+ \stackrel{\text{def}}{=} L^*L$, as usual)? As each input string $\boxed{\ }^{m}\boxed{\mathsf{P}p, q}^{n}$ can move at most n p's into $\Box^m$, there can be no more $\boxed{p}$'s than $\boxed{q}$'s in an output string, and we have

$$\{\mathsf{s}' \mid (\exists \mathsf{s} \in \boxed{\ }^{*}\boxed{\mathsf{P}p, q}^{*})\ \mathsf{s}\ \mathrm{tc}(\hat{R})\ \mathsf{s}'\} \cap \boxed{p}^{+}\boxed{q}^{+} = \{\boxed{p}^{m}\boxed{q}^{n} \mid 1 \leq m \leq n\}$$

which is non-regular. It follows that $\mathrm{tc}(\hat{R})$ cannot be a regular relation since the languages $\boxed{p}^{+}\boxed{q}^{+}$ and $\boxed{\ }^{*}\boxed{\mathsf{P}p, q}^{*}$ are regular. By contrast, constraints applied with $\trianglerighteq$-minimization and *unpad* intersect with $\boxed{p}^{+}\boxed{q}^{+}$ to give the regular sublanguage $\boxed{p}\boxed{q}^{+}$.

3 Conclusion

Regular relations $\mathscr{R}[\varphi]$ were defined for various absolute LTL formulas φ (collected in the set P_+) from regular languages $\mathscr{A}[\varphi]$ satisfying constraints $\mathscr{C}(\varphi)$ that essentially reformulate well-known LTL tableau rules in finite-state terms. A theorem was isolated at the end of Section 2.3 asserting that $\mathscr{A}[\varphi]$ encodes satisfaction $\models$ relative

where V_t is V shifted to the right by t

$$V_t \stackrel{\text{def}}{=} \{(t' - t, p) \mid V(t', p)\}.$$

From the perspective of Hybrid Logic (Areces and ten Cate 2007), LTL lacks a nominal marking out the position 0.

to arbitrary relations $V \subseteq \mathbb{Z} \times P$ over the full set $\mathbb{Z}$ of integers. The theorem is not surprising, but introduces machinery that is, I believe, of interest beyond the theorem. That said, the theorem does raise further questions about the relationship with LTL. Can we apply the about sets $\mathscr{A}[\varphi]$ to capture LTL entailments $\vdash^{\mathrm{LTL}}$ between formulas in P_+? What about non-absolute LTL formulas such as $\mathsf{G}p$? We turn to these questions below, before briefly touching one of the points behind introducing regular relations $\mathscr{R}[\varphi]$—i.e., that of varying temporal granularity.

3.1 Entailments short of $\vdash^{\mathrm{LTL}}$

A problem with reducing LTL entailments

$$\varphi \vdash^{\mathrm{LTL}} \psi \iff^{\mathrm{def}} (\forall V \subseteq \mathbb{Z} \times P)\,(\forall t \in \mathbb{Z})\ V, t \models \varphi \text{ implies } V, t \models \psi$$

to $\mathscr{A}[\varphi] \sqsubseteq \mathscr{A}[\psi]$ is that a string in $\mathscr{A}[\varphi]$ may be too small to contain one in $\mathscr{A}[\psi]$. Accordingly, the entailment

$$\varphi \vdash_C \psi \iff C \cap \langle\sqsubseteq\rangle \mathscr{A}[\varphi] \ \subseteq\ \langle\sqsubseteq\rangle \mathscr{A}[\psi]$$

enlarges a string in $\mathscr{A}[\varphi]$ to one in C before searching for a (sub)string in $\mathscr{A}[\psi]$. Each string, however, has a bounded temporal span short of $\mathbb{Z}$, representable as a finite segment

$$[i, j] \stackrel{\mathrm{def}}{=} \{t \in \mathbb{Z} \mid i \le t \le j\}$$

for $i, j \in \mathbb{Z}$. In effect, $V, t \models \varphi$ is replaced by

$$s, t \models_i^j \varphi \quad \text{for} \quad s \subseteq [i, j] \times P \text{ and } t \in [i, j],$$

where for example,

$$s, t \models_i^j \mathit{next}(\varphi) \iff^{\mathrm{def}} t < j \text{ and } s, t+1 \models_i^j \varphi$$
$$s, t \models_i^j \mathit{previous}(\varphi) \iff^{\mathrm{def}} i < t \text{ and } s, t-1 \models_i^j \varphi$$

and

$$s, t \models_i^j \mathsf{G}\varphi \iff^{\mathrm{def}} (\forall t' \in [t, j])\ s, t' \models_i^j \varphi.$$

For non-negative integers m and n, let C_m^n be the constraint

$$C_m^n \stackrel{\mathrm{def}}{=} (\boxed{\mathit{now}} \stackrel{\mathsf{b}}{\Rightarrow} \square^m) \cap (\boxed{\mathit{now}} \stackrel{\mathsf{a}}{\Rightarrow} \square^n) \cap (\boxed{\mathit{now}}\ \square^*\ \boxed{\mathit{now}} \Rightarrow \emptyset)$$

requiring that there be at least m boxes before the evaluation time *now* and n boxes after, and at most one occurrence of *now*. Then for $\varphi, \psi \in P_+$,

$$\varphi \vdash_{C_m^n} \psi \iff (\forall i \le -m)(\forall j \ge n)(\forall s \subseteq [i, j] \times P) \quad s, 0 \models_i^j \varphi \ \text{ implies } \ s, 0 \models_i^j \psi \tag{21}$$

with *now* normalized to 0 to simplify notation. To build the negation $\overline{p}$ of an atomic proposition p into P, we intersect C_m^n with the constraint

$$(\square \Rightarrow \boxed{p} \mid \boxed{\overline{p}}) \cap (\boxed{p, \overline{p}} \Rightarrow \emptyset)$$

corresponding to a restriction in (21) to relations $s \subseteq [i, j] \times P$ satisfying

$$s(t, \overline{p}) \iff \text{not } s(t, p)$$

for $t \in [i, j]$. Observe that $next^n(p \vee \overline{p})$ holds at 0 for any such relation s,

$$s, 0 \models_i^j next^n(p \vee \overline{p})$$

but not necessarily $next^{n+1}(p \vee \overline{p})$. Similarly for $previous^m(p \vee \overline{p})$.

Stepping beyond absolute formulas, notice that the LTL entailment

$$\mathsf{G}p \vdash^{\mathrm{LTL}} \mathsf{G}\ next(p)$$

(saying 'if p is true now and forever more, then so is $next(p)$') does *not* go through for $\vdash_C$ (unless C precludes $\mathsf{G}p$), as

$$s, t \not\models_i^j \mathsf{G}\ next(p)$$

for all $s \subseteq [i, j] \times P$ and $t \in [i, j]$.

3.2 $\mathsf{G}p$ *and some other non-absolute formulas*

To say $\mathsf{G}p$ is not absolute is to deny that $\mathscr{R}[\mathsf{G}p]$ can be the $\mathscr{A}[\mathsf{G}p]$-restriction of $\sqsupseteq$, whatever $\mathscr{A}[\mathsf{G}p]$ might be. But if, in place of $\sqsubseteq$, we define a relation $\hat{R}$ by

$$\mathsf{s}\ \hat{R}\ \mathsf{s}' \overset{\text{def}}{\iff} (\exists \mathsf{s}'')\ \mathsf{s} \trianglerighteq \mathsf{s}''\mathsf{s}'$$

(sandwiched between $\sqsupseteq$ and $\trianglerighteq$) and put

$$\mathscr{A}[\mathsf{G}p] \overset{\text{def}}{=} \boxed{now, p \mid p}^*$$

then we can set $\mathscr{R}[\mathsf{G}p]$ to the $\mathscr{A}[\mathsf{G}p]$-restriction of $\hat{R}$

$$\mathsf{s}\ \mathscr{R}[\mathsf{G}p]\ \mathsf{s}' \overset{\text{def}}{\iff} \mathsf{s}\ \hat{R}\ \mathsf{s}' \text{ and } \mathsf{s}' \in \mathscr{A}[\mathsf{G}p].$$

Similarly, for the past counterpart H of G with the semantic interpretation

$$s, t \models_i^j \mathsf{H}\varphi \overset{\text{def}}{\iff} (\forall t' \in [i, t])\ s, t' \models_i^j \varphi$$

let

$$\mathscr{A}[\mathsf{H}p] \overset{\text{def}}{=} \boxed{p}^* \boxed{now, p}$$

and

$$\mathsf{s}\ \check{R}\ \mathsf{s}' \overset{\text{def}}{\iff} (\exists \mathsf{s}'')\ \mathsf{s} \trianglerighteq \mathsf{s}'\mathsf{s}''$$

for

$$\mathsf{s}\ \mathscr{R}[\mathsf{H}p]\ \mathsf{s}' \overset{\text{def}}{\iff} \mathsf{s}\ \check{R}\ \mathsf{s}' \text{ and } \mathsf{s}' \in \mathscr{A}[\mathsf{H}p].$$

Generalizing over $\mathsf{G}p$ and $\mathsf{H}p$, we can use $\trianglerighteq$ in place of $\sqsubseteq$, $\hat{R}$ or $\check{R}$, provided we pad $\mathscr{A}[\mathsf{G}p]$ to the left for

$$\mathscr{A}'[\mathsf{G}p] \overset{\text{def}}{=} \boxed{\ }^* \boxed{now, p \mid p}^*$$

and $\mathscr{A}[\mathsf{H}p]$ to the right for

$$\mathscr{A}'[\mathsf{H}p] \stackrel{\text{def}}{=} \boxed{p}^* \boxed{now, p \,\big|\,}^*$$

so that

$$\mathsf{s}\ \mathscr{R}[\varphi]\ \mathsf{s}' \iff \mathsf{s} \trianglerighteq \mathsf{s}' \text{ and } \mathsf{s}' \in \mathscr{A}'[\varphi]. \tag{22}$$

Under (22), $\mathscr{A}'[\varphi]$ need not be padded

$$\mathscr{A}'[\mathsf{H}p \wedge \mathsf{G}q] \stackrel{\text{def}}{=} \boxed{p}^* \boxed{now, p, q \,\big|\, q}^*$$

although for absolute formulas, the price of strengthening $\sqsupseteq$ to $\trianglerighteq$ is padding to the left and right[6]

$$\mathscr{A}'[\varphi] \stackrel{\text{def}}{=} \Box^* \mathscr{A}[\varphi] \Box^* \quad \text{for } \varphi \in P_+$$

(undoing the step from $\sqsupseteq$ to $\trianglerighteq$). Padding goes against the grain of defining $\mathscr{R}[\varphi]$ to get a handle on the temporal span of a φ-situation (useful, for example, for past tense reports; recall Section 1.4 above). On the other hand, it is easy enough to unpad $\mathscr{A}'[\varphi]$ for $\mathscr{A}[\varphi]$.

3.3 From temporal span to grain

If the image of $\mathscr{R}[\varphi]$ is unpadded, then we can replace $\sqsupseteq$ by $\sqsupseteq_\circ$ inasmuch as

$$\mathsf{s} \sqsupseteq \mathsf{s}' \iff \mathsf{s} \sqsupseteq_\circ \mathsf{s}' \quad \text{if } \mathit{unpad}(\mathsf{s}') = \mathsf{s}',$$

where

$$\mathsf{s} \sqsupseteq_\circ \mathsf{s}' \stackrel{\text{def}}{\iff} \mathsf{s}\ \mathsf{has\text{-}factor}; \trianglerighteq \mathsf{s}'$$

and factors are suffixes of prefixes (equivalently, prefixes of suffixes)

$$\begin{aligned} \mathsf{s}\ \mathsf{has\text{-}factor}\ \mathsf{s}' &\iff \mathsf{s}\ \mathsf{has\text{-}prefix}; \mathsf{has\text{-}suffix}\ \mathsf{s}' \\ &\iff \mathsf{s}\ \mathsf{has\text{-}suffix}; \mathsf{has\text{-}prefix}\ \mathsf{s}' \end{aligned}$$

and a suffix is a tail end

$$\mathsf{s}\ \mathsf{has\text{-}suffix}\ \mathsf{s}' \stackrel{\text{def}}{\iff} (\exists \mathsf{s}'')\ \mathsf{s} = \mathsf{s}''\mathsf{s}'$$

and a prefix is an initial fragment

$$\mathsf{s}\ \mathsf{has\text{-}prefix}\ \mathsf{s}' \stackrel{\text{def}}{\iff} (\exists \mathsf{s}'')\ \mathsf{s} = \mathsf{s}'\mathsf{s}''.$$

The 'in' relation $\hat{R}$ for $\mathsf{G}p$ replaces has-factor by has-suffix

$$\mathsf{s}\ \hat{R}\ \mathsf{s}' \stackrel{\text{def}}{\iff} \mathsf{s}\ \mathsf{has\text{-}suffix}; \trianglerighteq \mathsf{s}'$$

while the 'in' relation $\check{R}$ for $\mathsf{H}p$ uses has-prefix

$$\mathsf{s}\ \check{R}\ \mathsf{s}' \stackrel{\text{def}}{\iff} \mathsf{s}\ \mathsf{has\text{-}prefix}; \trianglerighteq \mathsf{s}'.$$

[6] This is exactly analogous to the transformation of a stringwise representation L to the pathwise representation $\Box^* L \Box^*$ in Fernando (2008b).

Table 4. *Comparing information content modulo* π.

π-equivalent	$\mathsf{s} \equiv_\pi \mathsf{s}' \iff \pi(\mathsf{s}) = \pi(\mathsf{s}')$
π-contain	$\mathsf{s} \geq_\pi \mathsf{s}' \iff (\exists \mathsf{s}_\circ \equiv_\pi \mathsf{s})(\exists \mathsf{s}'_\circ \equiv_\pi \mathsf{s}')\ \mathsf{s}_\circ \trianglerighteq \mathsf{s}'_\circ$

That said, an advantage of the formulation $\sqsupseteq$ over $\sqsupseteq_\circ$

$$\mathsf{s} \sqsupseteq \mathsf{s}' \stackrel{\text{def}}{\iff} (\exists \mathsf{s}_\circ \approx \mathsf{s})(\exists \mathsf{s}'_\circ \approx \mathsf{s}')\ \mathsf{s}_\circ \trianglerighteq \mathsf{s}'_\circ$$

(where $\mathsf{s}_\circ \approx \mathsf{s} \stackrel{\text{def}}{\iff} \mathit{unpad}(\mathsf{s}_\circ) = \mathit{unpad}(\mathsf{s})$) is that it generalizes from *unpad* to an arbitrary function π (Table 4).

An example of π suggested by the slogan 'no time without change' (Kamp and Reyle 1993: 674) reduces all adjacent identical boxes $\alpha\alpha^n$ to one α. More precisely, the *block compression* $bc(\mathsf{s})$ *of* a string s is given by

$$bc(\mathsf{s}) \stackrel{\text{def}}{=} \begin{cases} bc(\alpha\mathsf{s}') & \text{if } \mathsf{s} = \alpha\alpha\mathsf{s}' \\ \alpha bc(\beta\mathsf{s}') & \text{if } \mathsf{s} = \alpha\beta\mathsf{s}' \text{ with } \alpha \neq \beta \\ \mathsf{s} & \text{otherwise (length}(\mathsf{s}) \leq 1) \end{cases}$$

so that, for example,

$$bc(\boxed{p,q}\boxed{p,q}\boxed{p,q}\boxed{p,q,r}\boxed{p,q,r}\boxed{p,q}) = \boxed{p,q}\boxed{p,q,r}\boxed{p,q}.$$

For a finite-state transducer computing bc, let the set of states be the alphabet plus the initial state 0, with transitions

$$0 \stackrel{\alpha:\alpha}{\to} \alpha \quad \text{for every symbol } \alpha$$
$$\alpha \stackrel{\beta:\beta}{\to} \beta \quad \text{for any two distinct symbols } \alpha, \beta$$

(to remember the symbol last seen)

$$\alpha \stackrel{\alpha:\epsilon}{\to} \alpha \quad \text{for every symbol } \alpha$$

(to remove duplications) and all states final (accepting). If a string s is construed as a sequence of observations, the effect of $bc(\mathsf{s})$ is to keep time still if no change (from the previous box) is observed. Exactly what observations are allowed is the crucial issue of granularity, which amounts in the present context to what fluents may occur in a symbol α. The LTL connective *previous* makes each tick of the clock $\mathbb{Z}$ count as an observation, thereby neutralizing bc. For instance, the string $\boxed{p}\boxed{p}\boxed{p}$ is treated in LTL as

$$\boxed{p}\boxed{p, previous(p)}\boxed{p, previous(previous(p))}$$

which bc leaves unchanged, whereas $bc(\boxed{p}\boxed{p}\boxed{p}) = \boxed{p}$. Evidently, we must apply block compression on a string only if it records all information of interest explicitly as fluents. As our interests may change, it is useful to define for any set $X \subseteq \Phi$ of fluents, a function ρ_X on strings that restricts the observations to X by intersecting componentwise with X

$$\rho_X(\alpha_1\alpha_2\cdots\alpha_n) \stackrel{\text{def}}{=} (\alpha_1 \cap X)(\alpha_2 \cap X)\cdots(\alpha_n \cap X).$$

For example, if $X = \{\mathsf{Jan}, \mathsf{Feb}, \ldots \mathsf{Dec}\}$, ρ_X projects the string

$$\boxed{\mathsf{Jan,d1}}\boxed{\mathsf{Jan,d2}}\cdots\boxed{\mathsf{Feb,d29}}\boxed{\mathsf{Mar,d1}}\cdots\boxed{\mathsf{Dec,d31}}$$

(over the 43 fluents Jan, Feb, … Dec, d1, d2, …, d31) representing a leap year to

$$\boxed{\mathsf{Jan}}^{31}\boxed{\mathsf{Feb}}^{29}\boxed{\mathsf{Mar}}^{31}\cdots\boxed{\mathsf{Dec}}^{31}$$

which $b\alpha$ maps to the string

$$\boxed{\mathsf{Jan}}\boxed{\mathsf{Feb}}\boxed{\mathsf{Mar}}\cdots\boxed{\mathsf{Dec}}$$

of length 12. Composing $\rho_X ; b\alpha$ further with *unpad* gives regular relations

$$\pi_X(\mathsf{s}) \stackrel{\text{def}}{=} \mathit{unpad}(b\alpha(\rho_X(\mathsf{s})))$$

relative to which event structures (in the sense of Kamp and Reyle 1993) can be constructed as inverse (projective) limits over finite sets X of fluents (Fernando 2007, 2010). Integrating calendar expressions with temporal propositions, the basic idea is to work with strings that have no more and no less information than what is given. Natural language descriptions do not, as a rule, mention a clock, and rarely the clock $\mathbb{Z}$ (however natural that may be for LTL applications).

Acknowledgments

I thank my referees for their critical and constructive comments, and Anssi Yli-Jyrä for his work as editor.

References

Areces, C., and ten Cate, B. 2007. Hybrid logics. In P. Blackburn, F. Wolter, and J. van Benthem (eds.), *Handbook of Modal Logic*, pp. 821–68. Amsterdam, The Netherlands: Elsevier.

Barwise, J., and Perry, J. 1983. *Situations and Attitudes*. Cambridge, MA, USA: The MIT Press.

Beesley, K. R., and Karttunen, L. 2003. *Finite State Morphology*. Stanford, CA, USA: CSLI Publications.

Bouajjani, A., Jonsson, B., Nilsson, M., and Touili, T. 2000. Regular model checking. In *Computer Aided Verification*, pp. 403–18. LNCS 1855. Berlin, Germany: Springer-Verlag.

Brink, C., Britz, K., and Schmidt, R. 1994. Peirce algebras. *Formal Aspects of Computing* **6**(3):339–58.

Condoravdi, C. 2009. Punctual *until* as a scalar NPI. In K. Hanson, and S. Inkelas (eds.), *The Nature of the Word: Studies in Honor of Paul Kiparsky*, pp. 631–53. Cambridge, MA, USA: The MIT Press.

Davidson, D. 1967. The logical form of action sentences. In N. Rescher (ed.), *The Logic of Decision and Action*, pp. 81–95. Pittsburgh, PA, USA: University of Pittsburgh Press.

Emerson, E. A. 1992. Temporal and modal logic. In J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, pp. 995–1072. Cambridge, MA, USA: The MIT Press.

Fernando, T. 2004. A finite-state approach to events in natural language semantics. *Journal of Logic and Computation* **14**(1): 79–92.

Fernando, T. 2007. Observing events and situations in time. *Linguistics and Philosophy* **30**(5): 527–50.

Fernando, T. 2008a. Branching from inertia worlds. *Journal of Semantics* **25**(3): 321–344.
Fernando, T. 2008b. Temporal propositions as regular languages. In T. Hanneforth, and K-M. Würzner (eds.), *Finite-State Methods and Natural Language Processing, 6th International Workshop*, pp. 132–48. Potsdam, Germany: Universitätsverlag Potsdam.
Fernando, T. 2009a. Situations in LTL as strings. *Information and Computation* **207** (10): 980–99.
Fernando, T. 2009b. Situations as indices and as denotations. *Linguistics and Philosophy* **32**(2): 185–206.
Fernando, T. 2010. Constructing situations and time. *Journal of Philosophical Logic*, doi:10.1007/s10992-010-9155-1.
Harel, D., Kozen, D., and Tiuryn, J. 2000. *Dynamic Logic*. Cambridge, MA, USA: The MIT Press.
Kamp, H., and Reyle, U. 1993. *From Discourse to Logic*. Dordrecht: Kluwer.
Karttunen, L. 1974 Until. In *Papers from the Tenth Regional Meeting of the Chicago Linguistic Society*, Chicago, pp. 284–97.
Karttunen, L. 2005 The Yale Shooting Problem. `http://www.stanford.edu/~laurik/fsmbook/examples/YaleShooting.html`
McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In M. Meltzer, and D. Michie (eds.), *Machine Intelligence 4*, pp. 463–502. Edinburgh, UK: Edinburgh University Press.
Mulligan, K., Simons, P., and Smith, B. 1984. Truth-makers. *Philosophy and Phenomenological Research* **44**: 287–321.
Niemi, J., and Koskenniemi, K. 2009. Representing and combining calendar information by using finite-state transducers. In J. Piskorski, B. Watson, and A. Yli-Jyrä (eds.), *Finite-State Methods and Natural Language Processing: Post-proceedings of the 7th International Workshop FSMNLP 2008*, pp. 122–33. Amsterdam, The Netherlands: IOS Press.
Reichenbach, H. 1947. *Elements of Symbolic Logic*. London, UK: Macmillan.
Vardi, M. Y. 2007. Automata-theoretic techniques for temporal reasoning. In P. Blackburn, F. Wolter, and J. van Benthem (eds.), *Handbook of Modal Logic*, pp. 971–89. Amsterdam, The Netherlands: Elsevier.