Types from Frames as Finite Automata

Tim Fernando(✉)

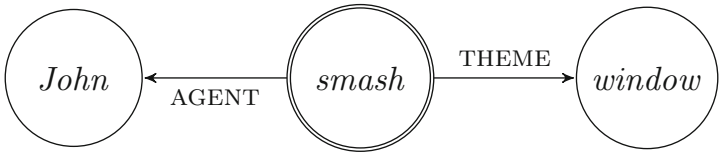
Trinity College Dublin, Dublin, Ireland
tim.fernando@cs.tcd.ie

Abstract. An approach to frame semantics is built on a conception of frames as finite automata, observed through the strings they accept. An institution (in the sense of Goguen and Burstall) is formed where these strings can be refined or coarsened to picture processes at various bounded granularities, with transitions given by Brzozowski derivatives.

Keywords: Frame · Finite automaton · Trace · Derivative · Institution

1 Introduction

A proposal for frame semantics recently put forward in Muskens (2013) analyzes a frame of the sort studied by Barsalou (1999); Löbner (2014), and Petersen and Osswald (2014) (not to forget Fillmore 1982) as a *fact* in a *data lattice* $\langle \mathfrak{F}, \circ, 0 \rangle$ with zero $0 \in \mathfrak{F}$ and meet $\circ : (\mathfrak{F} \times \mathfrak{F}) \to \mathfrak{F}$ (Veltman 1985). A frame such as



is analyzed, relative to any three entities e, x and y, as the $\circ$-combination

$$smash\, e \circ \text{AGENT}\, ex \circ John\, x \circ \text{THEME}\, ey \circ window\, y$$

of five facts, $smash\, e$, $\text{AGENT}\, ex$, $John\, x$, $\text{THEME}\, ey$, and $window\, y$. In general, any fact $g \in \mathfrak{F}$ induces a function $[g]$ from facts to one of three truth values, $\mathbf{t}$, $\mathbf{f}$ and $\mathbf{n}$, such that for all $f \in \mathfrak{F} - \{0\}$,

$$[g](f) \quad = \quad \begin{cases} \mathbf{t} & \text{if } f \circ g = f \text{ (i.e., } f \text{ incorporates } g\text{)} \\ \mathbf{f} & \text{if } f \circ g = 0 \text{ (i.e., } f \text{ and } g \text{ are incompatible)} \\ \mathbf{n} & \text{otherwise.} \end{cases}$$

Functions such as $[g]$ from $\mathfrak{F}$ to $\{\mathbf{t}, \mathbf{f}, \mathbf{n}\}$ are what Muskens calls *propositions*, breaking from possible worlds semantics in replacing possible worlds with facts, and adding a third truth value, $\mathbf{n}$, for a gap between truth and falsity. Sentences are interpreted as propositions, assembled compositionally from an interpretation of words as λ-abstracts of propositions, as in

$$\mathsf{smash} \quad = \quad \lambda y x \lambda f \exists e.[smash\, e \circ \text{AGENT}\, ex \circ \text{THEME}\, ey] f. \tag{1}$$

Muskens attaches significance to the separation of facts from propositions. Identifying frames with facts, he declares

A. Foret et al. (Eds.): FG 2015/2016, LNCS 9804, pp. 19–40, 2016.
DOI: 10.1007/978-3-662-53042-9_2

> I reject the idea (defended in Barsalou (1999), who explicitly discusses frame representations of negation, disjunction, and universal quantification) that all natural language meaning can profitably be represented with the help of frames

(page 176).

One way to evaluate Muskens' proposal is by comparing it with others. An alternative to existentially quantifying the event e in (1) is λ-abstraction, as in the analysis of sortal frames in Petersen and Osswald (2014), with

$$\lambda e.\ \mathit{smash}'(e) \wedge \mathit{animate}'(\textsc{agent}'(e)) \wedge \mathit{concrete}'(\textsc{theme}'(e)) \tag{2}$$

for the typed feature structure (a), or, to bring the example closer to (1),

$$\lambda e.\ \mathit{smash}'(e) \wedge \mathit{all}'(\textsc{agent}'(e)) \wedge \mathit{all}'(\textsc{theme}'(e)) \tag{3}$$

for the typed feature structure (b) over a vacuous all-encompassing type *all*.[1]

(a) $\begin{bmatrix} \mathit{smash} & \\ \textsc{agent} & \mathit{animate} \\ \textsc{theme} & \mathit{concrete} \end{bmatrix}$ (b) $\begin{bmatrix} \mathit{smash} & \\ \textsc{agent} & \mathit{all} \\ \textsc{theme} & \mathit{all} \end{bmatrix}$ (c) $\begin{bmatrix} \mathit{smash} \\ \textsc{agent} \\ \textsc{theme} \end{bmatrix}$

It is understood in both (2) and (3) that e is in the domain of the partial functions $\textsc{agent}'$ and $\textsc{theme}'$, making the terms $\textsc{agent}'(e)$ and $\textsc{theme}'(e)$ well-defined. We will simplify (b) shortly to (c), but before dropping *all*, let us use it to illustrate how to express the definedness presuppositions in (2) and (3) under the approach of Cooper (2012). To model a context, Cooper uses a record such as (d), which is of type (e) assuming *all* encompasses all.

(d) $\begin{bmatrix} \textsc{agent} = x \\ \textsc{theme} = y \end{bmatrix}$ (e) $\begin{bmatrix} \textsc{agent} : \mathit{all} \\ \textsc{theme} : \mathit{all} \end{bmatrix}$ (f) $\begin{bmatrix} p_1 : \mathit{smash}(r) \\ p_2 : \mathit{animate}(r.\textsc{agent}) \\ p_3 : \mathit{concrete}(r.\textsc{theme}) \end{bmatrix}$

Now, if *bg* is the record type (e), and φ is the type (f) dependent on a record r of type *bg*, we can form the function

$$(\lambda r : \mathit{bg})\ \varphi \tag{4}$$

mapping a record r of type *bg* to the record type φ. (4) serves as Cooper's meaning function with

- domain *bg* (for background) encoding the definedness presuppositions, and
- record type φ replacing what a Montagovian would have as a truth value.

Compared to the prefix λyx in Muskens' (1), the prefix $(\lambda r : \mathit{bg})$ in (4) provides not just the parameters x and y but the information that they are the agent and theme components of a record r, around which abstraction is centralized in accordance with the methodological assumption

[1] This introductory section presupposes some familiarity with the literature, but is followed by sections that proceed in a more careful manner, without relying on a full understanding of the Introduction.

(†) components are extracted from a single node associated with the frame.

The number of components may be open-ended, as argued famously for events in Davidson (1967)

Jones did it slowly, deliberately, in the bathroom with a knife, at midnight

(page 81). Under pressure from multiple components, the assumption (†) is relaxed for non-sortal frames in Löbner (2014) and Petersen and Osswald (2014), with the latter resorting to ϵ-terms (page 249) and ι-terms (page 252). It is, however, possible to maintain (†) by adding attributes that extend the central node to incorporate the required components (making ϵ- and ι-terms unnecessary). At stake in upholding (†) is a record type approach, a finite-state fragment of which is the subject of the present paper.

In Cooper (2012), record types are part of a rich system TTR of types with function spaces going well beyond finite-state methods. Viewing frames as finite automata — bottom-dwellers in the Chomsky hierarchy — certainly leads us back to Muskens' contention that frames cannot capture all natural language meaning. But while any *single* finite automaton is bound to fall short, much can be done with many automata. Or so the present paper argues. Very briefly, the idea is to reduce the matrices (a)–(c) to the sets (a)′–(c)′ of strings over the alphabet $\{smash, \textsc{agent}, \textsc{theme}, animate, concrete, all\}$, and to represent the typing in (g) by the matrix (h) that is reduced to the language (h)′ over an expansion of the alphabet with symbols a_x and a_y for x and y respectively.

(a)′ $\{smash,\ \textsc{agent}\ animate,\ \textsc{theme}\ concrete\}$
(b)′ $\{smash,\ \textsc{agent}\ all,\ \textsc{theme}\ all\}$
(c)′ $\{smash,\ \textsc{agent},\ \textsc{theme}\}$

(g) $\begin{bmatrix} smash \\ \textsc{agent} = x \\ \textsc{theme} = y \end{bmatrix} : \begin{bmatrix} smash \\ \textsc{agent} : animate \\ \textsc{theme} : concrete \end{bmatrix}$ (h) $\begin{bmatrix} smash & \\ \textsc{agent} & \begin{bmatrix} animate \\ a_x \end{bmatrix} \\ \textsc{theme} & \begin{bmatrix} concrete \\ a_y \end{bmatrix} \end{bmatrix}$

(h)′ $\{smash,\ \textsc{agent}\ a_x,\ \textsc{theme}\ a_y\} \cup \{agent\ animate,\ \textsc{theme}\ concrete\}$

To interpret the strings in the sets (a)′, (b)′, (c)′ and (h)′, we assume every symbol a in the string alphabet Σ is interpreted as a (partial) function $[\![a]\!]$, which we extend to strings $s \in \Sigma^*$, setting $[\![\epsilon]\!]$ to the identity, and $[\![sa]\!]$ to the sequential composition $\lambda x.[\![a]\!]([\![s]\!](x))$. Then in place of the λ-expressions (1) to (4), a language L is interpreted as the intersection

$$\bigcap_{s \in L} domain([\![s]\!]) \tag{5}$$

of the domains of the interpretations of strings in L. It is customary to present the interpretations $[\![a]\!]$ model-theoretically (as in the case of the interpretation $\textsc{agent}'$ of $\textsc{agent}$ in (2)), making the interpretations $[\![s]\!]$ and (5) model-theoretic.

But as will become clear below, the functions $[\![\alpha]\!]$ can also be construed as the α-labeled transitions in a finite automaton. The ultimate objective of the present work is to link frames to the finite-state perspective on events in Fernando (2015) (the slogan behind the bias for finite-state methods being *less is more*), as well as to more wide ranging themes of "semantics in flux" in Cooper (2012), and "natural languages as collections of resources" in Cooper and Ranta (2008).

The intersection (5) approximates the image $\{r : bg \mid \varphi\}$ of Cooper's meaning function $(\lambda r : bg)\varphi$ but falls short of maintaining the careful separation that $(\lambda r : bg)\varphi$ makes between the presuppositions bg and the dependent record type φ. That separation is recreated below within what is called an *institution* in Goguen and Burstall (1992), with bg formulated as a signature and φ as a sentence of that signature. Clarifying this formulation is largely what the remainder of the present paper is about, which consists of three sections, plus a conclusion. The point of departure of Sect. 2 is the determinism of a frame — the property that for every state (or node) q and every label a on an arc (or edge), there is at most one arc from q labeled a. Based on determinism and building on Hennessy and Milner (1985), Sect. 2 reduces a state q to a set of strings of labels (i.e., a language). This reduction is tested against states as types and states as particulars in Sect. 3. To ensure the languages serving as states are accepted by finite automata (i.e., regular languages), Sect. 4 works with various finite sets Σ of labels. The sets Σ are paired with record types for signatures, around which approximations are structured following Goguen and Burstall (1992).

One further word about the scope of the present work before proceeding. Functions and λ's are commonly taken for granted in a compositional syntax/semantics interface, yet another significant proposal for which is detailed in Kallmeyer and Osswald (2013) using Lexicalized Tree Adjoining Grammar (distinct from frames with a first-order formulation in Sects. 3.3.3–3.3.4 there compatible with (5) above[2]). The present paper steers clear of any choice of a syntactic formalism, making no claim of completeness in focusing (modestly) on types from frames as finite automata.

2 Deterministic Systems and Languages

Fix a (possibly infinite) set A of labels. An A-*deterministic system* is a partial function $\delta : Q \times \mathsf{A} \rightharpoonup Q$ from pairs $(q, a) \in Q \times \mathsf{A}$ to elements of Q, called states (of which there may or may not be infinitely many). Let ϵ be the null string (of length 0) and for any state $q \in Q$, let $\delta_q : \mathsf{A}^* \rightharpoonup Q$ be the partial Q-valued function from strings over the alphabet A that repeatedly applies δ starting at q; more precisely, δ_q is the $\subseteq$-least set P of pairs such that

(i) $(\epsilon, q) \in P$, and
(ii) $(sa, \delta(q', a)) \in P$ whenever $(s, q') \in P$ and $(q', a) \in \mathit{domain}(\delta)$.

[2] The compatibility here becomes obvious if the moves described in footnote 5 of page 281 in Kallmeyer and Osswald (2013) are made, and a root added with attributes to the multiple base nodes.

For example,

$$aa' \in \mathit{domain}(\delta_q) \iff (q,a) \in \mathit{domain}(\delta) \text{ and } a' \in \mathit{domain}(\delta_{\delta(q,a)}).$$

The partial functions δ_q determine

$$\mathit{transitions}\, q \xrightarrow{s} \delta_q(s) \text{ whenever } s \in \mathit{domain}(\delta_q)$$

which we can also read as

$$s\text{-}\mathit{components}\, \delta_q(s) \text{ of } q, \text{ for all } s \in \mathit{domain}(\delta_q).$$

The labels in A may correspondingly be regarded as acts or as attributes. In either case, there is, we will see, a useful sense in which the language $\mathit{domain}(\delta_q)$ over A holds just about all the A-deterministic system δ has to say about q. An element of $\mathit{domain}(\delta_q)$ is called a *trace of* q (*from* δ), and henceforth, we write $\mathit{trace}_\delta(q)$ interchangably with $\mathit{domain}(\delta_q)$.

2.1 Satisfaction and Traces

Given a set A of labels, the set Φ_A of (A-modal) formulas φ is generated

$$\varphi \ ::= \ \top \mid \neg\varphi \mid \varphi \wedge \varphi' \mid \langle a \rangle \varphi$$

from a tautology $\top$, negation $\neg$, conjunction $\wedge$, and modal operators $\langle a \rangle$ with labels $a \in \mathsf{A}$ (Hennessy and Milner 1985). We interpret a formula $\varphi \in \Phi_\mathsf{A}$ relative to an A-deterministic system $\delta : Q \times \mathsf{A} \rightharpoonup Q$ and state $q \in Q$ via a satisfaction relation $\models$ in the usual way, with (keeping δ implicit in the background)

$$q \models \top,$$

'not' $\neg$

$$q \models \neg\varphi \iff \text{not } q \models \varphi,$$

'and' $\wedge$

$$q \models \varphi \wedge \varphi' \iff q \models \varphi \text{ and } q \models \varphi'$$

and the accessibility relation $\{(q, \delta_q(a)) \mid q \in Q \text{ and } a \in \mathit{domain}(\delta_q)\}$ for $\langle a \rangle$

$$q \models \langle a \rangle \varphi \iff a \in \mathit{domain}(\delta_q) \text{ and } \delta_q(a) \models \varphi$$

It is not difficult to see that the set

$$\Phi_\mathsf{A}(q) \ := \ \{\varphi \in \Phi_\mathsf{A} \mid q \models \varphi\}$$

of formulas $\models$-satisfied by q depends precisely on $\mathit{domain}(\delta_q)$. That is, recalling that $\mathit{trace}_\delta(q)$ is $\mathit{domain}(\delta_q)$, the following conditions, (a) and (b), are equivalent for all states $q, q' \in Q$.

(a) $trace_\delta(q) = trace_\delta(q')$
(b) $\Phi_{\mathsf{A}}(q) = \Phi_{\mathsf{A}}(q')$

Let us write $q \sim q'$ if (a), or equivalently (b), holds,[3] and pronounce $\sim$ *trace equivalence*.

2.2 Identity of Indiscernibles and states as Languages

Identity of indiscernibles (also known as Leibniz's law, and mentioned in Osswald 1999, invoking Quine) can be understood against the set A of attributes as the requirement on δ that distinct pairs q, q' of states (in Q) *not* be trace equivalent

$$q \neq q' \implies q \not\sim q'.$$

Basing discernibility on formulas $\varphi \in \Phi_{\mathsf{A}}$, we say φ *differentiates* q *from* q' if $q \models \varphi$ but not $q' \models \varphi$. It follows that

$$\varphi \text{ differentiates } q \text{ from } q' \iff \neg\varphi \text{ differentiates } q' \text{ from } q$$

and

$$q \sim q' \iff \textit{no} \text{ formula in } \Phi_{\mathsf{A}} \text{ differentiates } q \text{ from } q'.$$

We can replace formulas by attributes and make differentiation symmetric, by agreeing that a label a *differentiates* q *from* q' if (exactly) one of the following holds

(i) $a \in trace_\delta(q) - trace_\delta(q')$
(ii) $a \in trace_\delta(q') - trace_\delta(q)$
(iii) $a \in trace_\delta(q) \cap trace_\delta(q')$ and $\Phi_{\mathsf{A}}(\delta_q(a)) \neq \Phi_{\mathsf{A}}(\delta_{q'}(a))$

In the case of (i) and (ii), we can see $q \not\sim q'$ already at a, whereas (iii) digs deeper. Two other equivalent ways to say a differentiates q from q' are (a) and (b) below.

(a) a is a prefix of a string in the symmetric difference of trace sets

$$(trace_\delta(q) \cup trace_\delta(q')) - (trace_\delta(q) \cap trace_\delta(q'))$$

(b) there exists $\varphi \in \Phi_{\mathsf{A}}$ such that the formula $\langle a \rangle \varphi$ differentiates either q from q' or q' from q

The notion of an attribute $a \in \mathsf{A}$ differentiating q from q' generalizes straightforwardly to a string $a_1 \cdots a_n \in \mathsf{A}^+$ differentiating q from q'.

In fact, if we reduce a state q to the language $trace_\delta(q)$, the notions of differentiation above link up smoothly with derivatives of languages (Brzozowski 1964;

[3] Readers familiar with bisimulations will note that $\sim$ is the largest bisimulation (determinism being an extreme form of image-finiteness; Hennessy and Milner 1985).

Conway 1971; Rutten 1998, among others). Given a language L and a string s, the *s-derivative of* L is the set

$$L_s \;:=\; \{s' \mid ss' \in L\}$$

obtained from strings in L that begin with s, by stripping s off. Observe that for all $q \in Q$ and $s \in \mathit{trace}_\delta(q)$, if $L = \mathit{trace}_\delta(q)$ then the s-derivative of L corresponds to the s-component $\delta_q(s)$ of q

$$L_s \;=\; \mathit{trace}_\delta(\delta_q(s))$$

and L decomposes into its components

$$L \;=\; \epsilon + \sum_{a \in \mathsf{A}} aL_a. \tag{6}$$

The fact that ϵ belongs to $\mathit{trace}_\delta(q)$ reflects prefix-closure. More precisely, a language L is said to be *prefix-closed* if $s \in L$ whenever $sa \in L$. That is, L is prefix-closed iff $\mathit{prefix}(L) \subseteq L$, where the set $\mathit{prefix}(L)$ of prefixes in L

$$\mathit{prefix}(L) \;:=\; \{s \mid L_s \neq \emptyset\}$$

consists of all strings that induce non-empty derivatives. For any non-empty prefix-closed language L, we can form a deterministic system δ over the set

$$\{L_s \mid s \in L\}$$

of s-derivatives of L, for $s \in L$, including ϵ for $L_\epsilon = L = \mathit{trace}_\delta(L)$, where $\mathit{domain}(\delta)$ is defined to be $\{(L_s, a) \mid sa \in L\}$ with

$$\delta(L_s, a) \;:=\; L_{sa} \;\text{ whenever } sa \in L.$$

But what about languages that are not prefix-closed? Without the assumption that L is prefix-closed, we must adjust Eq. (6) to

$$L \;=\; o(L) + \sum_{a \in \mathsf{A}} aL_a$$

with ϵ replaced by $\emptyset$ in case $\epsilon \notin L$, using

$$o(L) \;:=\; \begin{cases} \epsilon & \text{if } \epsilon \in L \\ \emptyset & \text{otherwise} \end{cases}$$

(called the *constant part* or *output* of L in Conway 1971, page 41). Now, the chain of equivalences

$$a_1 \cdots a_n \in L \iff a_2 \cdots a_n \in L_{a_1} \iff \cdots \iff \epsilon \in L_{a_1 \cdots a_n}$$

means that L is accepted by the automaton with

(i) all s-derivatives of L as states (whether or not $s \in \mathit{prefix}(L)$)

$$Q \;:=\; \{L_s \mid s \in \mathsf{A}^*\}$$

(ii) s-derivatives L_s for $s \in L$ as *final* (accepting) states
(iii) transitions forming a total function $Q \times \mathsf{A} \to Q$ mapping (L_s, a) to L_{sa}

and initial state $L_\epsilon = L$ (e.g., Rutten 1998).

An alternative approach out of prefix closure (from deterministic systems) is to define for any label $a \in \mathsf{A}$ and language $L \subseteq \mathsf{A}^*$, the *a-coderivative of L* to be the set

$$_aL \;:=\; \{s \mid sa \in L\}$$

of strings that, with a attached at the end, belong to L. Observe that the a-coderivative of a prefix-closed language is not necessarily prefix-closed (in contrast to s-derivatives). Furthermore,

Fact 1. *Every language is the coderivative of a prefix-closed language.*

Fact 1 is easy to establish: given a language L, attach a symbol a not occurring in L to the end of L, and form $\mathit{prefix}(La)$ before taking the a-coderivative

$$_a\mathit{prefix}(La) \;=\; L.$$

An a-coderivative effectively builds in a notion of final state (lacking in a deterministic system δ) around not $o(L)$ but $o(L_a)$, checking if ϵ is in L_a, rather than L (i.e., $a \in L$, rather than $\epsilon \in L$). The idea of capturing a type of state through a label (such as a for a-coderivatives) is developed further next.

3 From Attribute Values to Types and Particulars

An A-deterministic system $\delta : Q \times \mathsf{A} \rightharpoonup Q$ assigns each state $q \in Q$ a set

$$\hat{\delta}(q) \;:=\; \{(a, \delta(q,a)) \mid a \in \mathsf{A} \cap \mathit{trace}_\delta(q)\}$$

of attribute value pairs (a, q') with values q' that can themselves be thought as sets $\hat{\delta}(q')$ of attribute values pairs. Much the same points in Sect. 2 could be made appealing to the modal logic(s) of attribute value structures (Blackburn 1993) instead of Hennessy and Milner (1985). But is the reduction of q to its trace set, $\mathit{trace}_\delta(q)$, compatible with intuitions about attribute value structures? Let us call a state q *δ-null* if $\hat{\delta}(q) = \emptyset$; i.e., $\mathit{trace}_\delta(q) = \{\epsilon\}$. Reducing a δ-null state q to $\mathit{trace}_\delta(q) = \{\epsilon\}$ lumps all δ-null states into one — which is problematic if distinct atomic values are treated as δ-null (Blackburn 1993). But what if equality with a fixed value were to count as a discerning attribute? Let us define a state q to be *δ-marked* if there is a label $a_q \in \mathsf{A}$ such that for all $q' \in Q$,

$$a_q \in \mathit{trace}_\delta(q') \iff q = q'.$$

Clearly, a δ-marked state q is trace equivalent only to itself. If a state q is *not* δ-marked, we can introduce a fresh label a_q (not in A) and form the $(\mathsf{A} \cup \{a_q\})$-deterministic system

$$\delta[q] \;:=\; \delta \cup \{(q, a_q, q)\}$$

with q $\delta[q]$-marked. To avoid infinite trace sets $\mathit{trace}_{\delta[q]}(q) \supseteq {a_q}^*$ (from loops (q, a_q, q)), we can instead fix a δ-null (or fresh) state $\surd$ and mark q in

$$\delta[q, \surd] \;:=\; \delta \cup \{(q, a_q, \surd)\}.$$

Marking a state is an extreme way to impose Leibniz's law. A more moderate alternative described below introduces types over states, adding constraints to pick out particulars.

3.1 Type-Attribute Specifications and Containment

To differentiate states in a set Q through subsets $Q_t \subseteq Q$ given by types $t \in T$ is to define a binary relation $v \subseteq Q \times T$ (known in Kripke semantics as a *T-valuation*) such that for all $q \in Q$ and $t \in T$

$$v(q, t) \iff q \in Q_t.$$

We can incorporate v into $\delta : Q \times \mathsf{A} \rightharpoonup Q$ by adding a fresh attribute a_t, for each $t \in T$, to A for the expanded attribute set

$$\mathsf{A}_T \;:=\; \mathsf{A} \cup \{a_t \mid t \in T\}$$

and forming either the A_T-deterministic system

$$\delta[v] \;:=\; \delta \cup \{(q, a_t, q) \mid t \in T \text{ and } v(q, t)\}$$

or, given a δ-null state $\surd$ outside $\bigcup_{t \in T} Q_t$, the A_T-deterministic system

$$\delta[v, \surd] \;:=\; \delta \cup \{(q, a_t, \surd) \mid t \in T \text{ and } v(q, t)\}.$$

In practice, a type t may be defined from other types, as in (a) below.

$$\text{(a)}\quad t = \begin{bmatrix} \mathit{smash} & : \top \\ \textsc{agent} & : \mathit{animate} \\ \textsc{theme} & : \mathit{concrete} \end{bmatrix} \qquad\qquad \text{(b)}\quad e = \begin{bmatrix} \mathit{smash} & = x \\ \textsc{agent} & = y \\ \textsc{theme} & = z \end{bmatrix}$$

The record e given by (b) is an instance of t just in case x, y and z are instances of the types $\top$, *animate* and *concrete* respectively

$$e : t \iff x : \top \text{ and } y : \mathit{animate} \text{ and } z : \mathit{concrete}.$$

It is natural to analyze t in (a) as the modal formula φ_t with three conjuncts

$$\varphi_t \;=\; \langle \mathit{smash} \rangle \top \wedge \langle \textsc{agent} \rangle \langle \mathit{animate} \rangle \top \wedge \langle \textsc{theme} \rangle \langle \mathit{concrete} \rangle \top$$

(implicitly analyzing $\top$, *animate* and *concrete* as $\top$, $\langle animate\rangle\top$ and $\langle animate\rangle\top$ respectively) and to associate e with the trace set

$$q(e) \;=\; \epsilon \;+\; smash\; q(x) \;+\; \textsc{agent}\; q(y) \;+\; \textsc{theme}\; q(z)$$

(given trace sets $q(x), q(y)$ and $q(z)$ for x, y and z respectively) for the reduction

$$\begin{aligned} e : t \;&\iff\; q(e) \models \varphi_t \\ &\iff\; animate \in q(y) \text{ and } concrete \in q(z). \end{aligned}$$

We can rewrite (a) as the A'-deterministic system

$$\tau' \;:=\; \{(t, smash, \top),\; (t, \textsc{agent}, animate),\; (t, \textsc{theme}, concrete)\}$$

over the attribute set

$$\mathsf{A}' \;:=\; \{smash, \textsc{agent}, \textsc{theme}\}$$

and state set

$$T' \;:=\; \{t, \top, animate, concrete\}$$

given by types. To apply τ' to a deterministic system δ with states given by tokens of types, the following notion will prove useful. A $(T, \mathsf{A}, \surd)$-*specification* is an A_T-deterministic system $\tau : T \times \mathsf{A}_T \rightharpoonup T$ where $\surd \in T$ is τ-null and each $t \in T - \{\surd\}$ is τ-marked, with for all $t' \in T$,

$$(t', a_t) \in domain(\tau) \;\iff\; t = t'$$

(the intuition being to express a type t as the modal formula $\langle a_t\rangle\top$). For τ' given above, we can form the $(T' \cup \{\surd\}, \mathsf{A}', \surd)$-specification

$$\tau' \;\cup\; \{(x, a_x, \surd) \mid x \in T'\}.$$

Let us agree that a set Ψ of modal formulas is *true in* δ if every formula in Ψ is satisfied, relative to δ, by every δ-state. The content of a $(T, \mathsf{A}, \surd)$-specification τ is given by the set

$$\begin{aligned} spec(\tau) \;:=\; &\{\langle a_t\rangle\top \supset \langle a\rangle\top \mid (t, a, \surd) \in \tau\} \cup \\ &\{\langle a_t\rangle\top \supset \langle a\rangle\langle a_{t'}\rangle\top \mid (t, a, t') \in \tau \text{ and } t' \neq \surd\} \end{aligned}$$

of formulas true in an A_T-deterministic system δ precisely if for every $(t, a, t') \in \tau$ and $q \in Q_t$,

$$a \in trace_\delta(q) \;\text{ and }\; \delta(q, a) \in Q_{t'}$$

where for every $t \in T - \{\surd\}$, Q_t is $\{q \in Q \mid a_t \in trace_\delta(q)\}$ and $Q_\surd = Q$. We can express membership in a trace set

$$s \in trace_\delta(q) \;\iff\; q \models_\delta \langle s\rangle\top$$

through formulas $\langle s\rangle\varphi$ defined by induction on s:

$$\langle\epsilon\rangle\varphi \;:=\; \varphi \;\text{ and }\; \langle as\rangle\varphi \;:=\; \langle a\rangle\langle s\rangle\varphi$$

so that for $a_1a_2\cdots a_n \in \mathsf{A}^n$,

$$\langle a_1a_2\cdots a_n\rangle\varphi \;=\; \langle a_1\rangle\langle a_2\rangle\cdots\langle a_n\rangle\varphi.$$

Given a language L, let us say q *δ-contains* L if $L \subseteq \mathit{trace}_\delta(q)$.

Fact 2. *For any $(T, \mathsf{A}, \surd)$-specification τ, $\mathit{spec}(\tau)$ is true in an A_T-deterministic system $\delta : Q\times\mathsf{A}_T \rightharpoonup Q$ iff for every $t \in T-\{\surd\}$ and $q \in Q$, q δ-contains $\mathit{trace}_\tau(t)$ whenever q δ-contains $\{a_t\}$.*

Under certain assumptions, $\mathit{trace}_\tau(t)$ is finite. More specifically, let $T_0 = \emptyset$ and for any integer $n \geq 0$,

$$T_{n+1} \;:=\; \{t \in T \mid \tau \cap\; (\{t\} \times \mathsf{A} \times T) \subseteq T \times \mathsf{A} \times T_n\}$$

(making $T_1 = \{\surd\}$). For each $t \in T_n$, $\mathit{trace}_\tau(t)$ is finite provided

($\star$) for all $t \in T$, $\{a \in \mathsf{A} \mid (t, a) \in \mathit{domain}(\tau)\}$ is finite.

Although ($\star$) and $T \subseteq \bigcup_n T_n$ can be expected of record types in Cooper (2012), notice that if $(t, a, t) \in \tau$ then $t \notin \bigcup_n T_n$ and $a^* \subseteq \mathit{trace}_\tau(t)$.

3.2 Terminal Attributes, $\diamond$ and Subtypes

Fact 2 reduces a $(T, \mathsf{A}, \surd)$-specification τ to its trace sets, $\mathit{trace}_\tau(t)$ for $t \in T - \{\surd\}$, unwinding $\mathit{spec}(\tau)$ to

$$\langle a_t\rangle\top \;\supset \bigwedge_{s\in \mathit{trace}_\tau(t)} \langle s\rangle\top \tag{7}$$

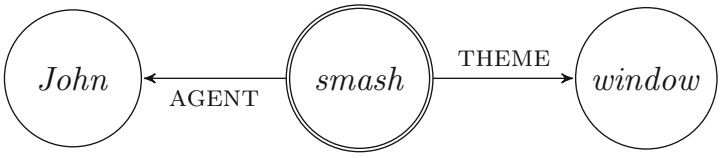
for $t \in T - \{\surd\}$ (where the conjunction might be infinite). The converse of (7) follows from $a_t \in \mathit{trace}_\tau(t)$. (7) has the form

$$\varphi_t \;\supset\; \varphi_{t[\tau]}$$

with antecedent $\langle a_t\rangle\top$ construed as a formula φ_t representing t, and consequent $\bigwedge_{s\in \mathit{trace}_\tau(t)}\langle s\rangle\top$ as a formula $\varphi_{t[\tau]}$ representing τ's conception of t. The attribute a_t often has the following property. Given an A-deterministic system δ, let us say an attribute $a \in \mathsf{A}$ is *δ-terminal* if the set

$$\mathit{Tml}_\mathsf{A}(a) \;:=\; \{\neg\langle sab\rangle\top \mid s \in \mathsf{A}^* \text{ and } b \in \mathsf{A}\}$$

of formulas is true in δ — i.e., for every δ-state q, string $s \in \mathsf{A}^*$ and attribute $b \in \mathsf{A}$, $sab \notin \mathit{trace}_\delta(q)$. It is natural to associate a frame such as



with an A-deterministic system δ in which labels on nodes (i.e., *John*, *smash* and *window*) are δ-terminal, while labels on arcs (AGENT, THEME) are not.

Next, we quantify away the strings s mentioned in $Tml_{\mathsf{A}}(a)$ for a useful modal operator $\Diamond$. Given an A-deterministic system $\delta : Q \times \mathsf{A} \rightharpoonup Q$, and states $q, q' \in Q$, we say q' is a *δ-component of q* and write $q \rightsquigarrow_\delta q'$, if q' is $\delta_q(s)$ for some string $s \in \mathsf{A}^*$. With the relation $\rightsquigarrow_\delta$, we extend satisfaction $\models$ to formulas $\Diamond\varphi$

$$q \models \Diamond\varphi \iff (\exists q')\ q \rightsquigarrow_\delta q' \text{ and } q' \models \varphi$$

making $\Diamond\varphi$ essentially the infinitary disjunction $\bigvee_{s\in\mathsf{A}^*}\langle s\rangle\varphi$ and its dual $\Box\varphi := \neg\Diamond\neg\varphi$ the infinitary conjunction $\bigwedge_{s\in\mathsf{A}^*}[s]\varphi$ (where $[s]\varphi := \neg\langle s\rangle\neg\varphi$). The extension preserves invariance under trace equivalence $\sim$

$$\text{whenever } q \sim q' \text{ and } q \models \Diamond\varphi,\ q' \models \Diamond\varphi.$$

That is, for the purpose of $\models$, we can reduce a state q to its trace set $trace_\delta(q)$. Accordingly, we collect all non-empty prefix-closed subsets of A^* in

$$Mod(\mathsf{A}) := \{prefix(L \cup \{\epsilon\}) \mid L \subseteq \mathsf{A}^*\}$$

(where "Mod" is for models) with the understanding that the transitions δ between $q, q' \in Mod(\mathsf{A})$ are given by derivatives

$$\delta(q, a) = q' \iff q_a = q'.$$

Given a set Ψ of modal formulas over A, we form the subset of $Mod(\mathsf{A})$ satisfying every formula of Ψ

$$Mod_{\mathsf{A}}(\Psi) := \{q \in Mod(\mathsf{A}) \mid (\forall\varphi \in \Psi)\ q \models \varphi\}$$

and say Ψ is A-*equivalent to* a set Ψ' of modal formulas over A if they have the same models

$$\Psi \equiv_{\mathsf{A}} \Psi' \iff Mod_{\mathsf{A}}(\Psi) = Mod_{\mathsf{A}}(\Psi').$$

Ψ can be strengthened to

$$\begin{aligned}\Box_{\mathsf{A}}(\Psi) &:= \{\neg\langle s\rangle\neg\varphi \mid s \in \mathsf{A}^* \text{ and } \varphi \in \Psi\}\\ &\equiv_{\mathsf{A}} \{\Box\varphi \mid \varphi \in \Psi\}\end{aligned}$$

requiring that every formula in Ψ holds in all components. Clearly,

$$Tml_{\mathsf{A}}(a) \equiv_{\mathsf{A}} \Box_{\mathsf{A}}(\{\neg\langle ab\rangle\top \mid b \in \mathsf{A}\}).$$

Given representations φ_t and φ_u of types t and u, we can express the subtyping $t \sqsubseteq u$ as the set

$$`t \sqsubseteq u' \;:=\; \{\neg\langle s\rangle(\varphi_t \wedge \neg\varphi_u) \mid s \in \mathsf{A}^*\}$$

of modal formulas denying the existence of components of type t but not u. Then '$t \sqsubseteq u$' requires $\varphi_t \supset \varphi_u$ of all components

$$`t \sqsubseteq u' \;\equiv_{\mathsf{A}}\; \Box(\varphi_t \supset \varphi_u)$$

bringing us back to the implication (7) above of the form $\varphi_t \supset \varphi_{\tau[t]}$. Inasmuch as an attribute a is represented by the formula $\langle a\rangle\top$, we can speak of a being contained in an attribute b, $a \sqsubseteq b$, when asserting

$$\Box(\langle a\rangle\top \supset \langle b\rangle\top).$$

3.3 Typings and Particulars

We can reduce a typing $p : t$ to a subtyping through the equivalence

$$p : t \iff \{p\} \sqsubseteq t$$

assuming we can make sense of the singleton type $\{p\}$. Given an A-deterministic system δ and a δ-state q, let us call a formula φ an (A, q)-*nominal* if q satisfies the set

$$\begin{aligned} \mathit{Nom}_{\mathsf{A}}(\varphi) \;&:=\; \{\neg(\langle s'\rangle(\varphi \wedge \langle s\rangle\top) \;\wedge\; \langle s''\rangle(\varphi \wedge \neg\langle s\rangle\top)) \mid s, s', s'' \in \mathsf{A}^*\} \\ &\equiv_{\mathsf{A}}\; \{\neg(\Diamond(\varphi \wedge \langle s\rangle\top) \;\wedge\; \Diamond(\varphi \wedge \neg\langle s\rangle\top)) \mid s \in \mathsf{A}^*\} \end{aligned}$$

of formulas that together say any two δ-components that δ-satisfy φ δ-contain the same languages over A. We can rephrase $\mathit{Nom}_{\mathsf{A}}(\varphi)$ as implications

$$\begin{aligned} \mathit{Nom}_{\mathsf{A}}(\varphi) \;&\equiv_{\mathsf{A}}\; \{\Diamond(\varphi \wedge \langle s\rangle\top) \supset \Box(\varphi \supset \langle s\rangle\top) \mid s \in \mathsf{A}^*\} \\ &\equiv_{\mathsf{A}}\; \{\Diamond(\varphi \wedge \psi) \supset \Box(\varphi \supset \psi) \mid \psi \in \Phi_{\mathsf{A}}\} \end{aligned}$$

making δ-components that δ-satisfy φ Φ_{A}-indistinguishable.[4]

Fact 3. *Let δ be an A-deterministic system, q be a δ-state and φ be an (A, q)-nominal.*

(i) For all δ-components q' and q'' of q,

$$q' \models \varphi \text{ and } q'' \models \varphi \;\; \textit{implies} \;\; q' \sim q''.$$

[4] $\mathit{Nom}_{\mathsf{A}}(\varphi)$ does the work of the scheme (Nom_N) for nominals given in Blackburn (1993), just as $\mathit{Tml}_{\mathsf{A}}(a)$ is analogous to the scheme (Term) there for instantiating atomic information at terminal nodes.

(ii) There is a δ-component of q that δ-satisfies φ and ψ

$$q \models \Diamond(\varphi \wedge \psi)$$

iff every δ-component of q satisfies $\varphi \supset \psi$ and some δ-component of q δ-satisfies φ

$$q \models \Box(\varphi \supset \psi) \ \wedge \ \Diamond\varphi.$$

Now, an (A, q)-*particular* is an (A, q)-nominal φ such that q has a δ-component that δ-satisfies φ — i.e., in addition to all formulas in $Nom_{\mathsf{A}}(\varphi)$, q satisfies $\Diamond\varphi$. We say (A, q) *verifies* $p : t$ if

$$\varphi_{\{p\}} \text{ is an } (\mathsf{A}, q)\text{-particular and } q \models \Box(\varphi_{\{p\}} \supset \varphi_t).$$

Part (ii) of Fact 3 says this is equivalent to $\varphi_{\{p\}}$ being an (A, q)-nominal and q having a δ-component that δ-satisfies $\varphi_{\{p\}} \wedge \varphi_t$. Note that if q is δ-marked by a_q then $\langle a_q\rangle\top$ is an (A, q')-nominal for all δ-states q'. The weaker notion of an (A, q)-nominal φ has the advantage over $\langle a_q\rangle\top$ that the set $Nom_{\mathsf{A}}(\varphi)$ of modal formulas allows the modal formulas satisfied by a state δ-satisfying $\varphi_{\{p\}}$ to vary with δ.[5] We can use the set $Nom_{\mathsf{A}}(\varphi) \cup \{\Diamond\varphi\}$ to lift the notion of an (A, q)-particular to that of a (A, Ψ)-*particular*, given a set Ψ of modal formulas, requiring that $Nom_{\mathsf{A}}(\varphi) \cup \{\Diamond\varphi\}$ follow from Ψ in that

$$Mod_{\mathsf{A}}(\Psi) \ \subseteq \ Mod_{\mathsf{A}}(Nom_{\mathsf{A}}(\varphi) \cup \{\Diamond\varphi\}).$$

Having described how a particular can be understood as a formula (or attribute a via $\langle a\rangle\top$), it is perhaps worth emphasizing that we need *not* understand particulars as formulas (or attributes). In the present context, particulars are first and foremost δ-states in an A-deterministic system δ. For any δ-state q, we can speak of types to which q belongs with*out* introducing an attribute that (relative to an extension of δ) represents q. What is important is that we build into A all the structure in particulars we wish to analyze.

4 Finite Approximations and Signatures

Throughout this section, we shall work with a set A of labels that is large enough so that we may assume trace equivalence $\sim$ is equality. Identity of indiscernibles in an A-deterministic system δ reduces a state q to $trace_\delta(q)$, allowing us to identify a state with a prefix-closed, non-empty subset of A^*. As huge as A can be, we can form the set $Fin(\mathsf{A})$ of finite subsets of A and use the equality

$$\mathsf{A}^* \ = \ \bigcup_{\Sigma \in Fin(\mathsf{A})} \Sigma^*$$

[5] By contrast, for a language q over A, all formulas in the set

$$\{\Box(\langle a_q\rangle\top \supset (\langle s\rangle\top \wedge \neg\langle s'\rangle\top)) \mid s \in q \text{ and } s' \in \mathsf{A}^* - q\}$$

must be satisfied for a_q to mark q.

to approach a language $L \subseteq \mathsf{A}^*$ via its intersections $L \cap \Sigma^*$ (for $\Sigma \in \mathit{Fin}(\mathsf{A})$)

$$L \;=\; \bigcup_{\Sigma \in \mathit{Fin}(\mathsf{A})} (L \cap \Sigma^*)$$

at the cost of bounding discernibility to Σ. For $X \in \mathit{Fin}(\mathsf{A}) \cup \{\mathsf{A}\}$, we define an *X-state* to be a non-empty, prefix-closed subset q of X^* on which transitions are given by derivatives

$$\delta(q, a) = q_a \quad \text{for } a \in \mathsf{A} \cap q$$

making $\delta_q(s) = q_s$ for $s \in q$. Henceforth, we put the notation δ aside, but track the parameter X, which we set to a finite subset Σ of A to pick out what is of particular interest.

Two subsections make up the present section. The first illustrates how the set A of labels can explode; the second how, after this explosion, to keep a grip on satisfaction $\models$ of formulas. We associate the formulas with sets $X \in \mathit{Fin}(\mathsf{A}) \cup \{\mathsf{A}\}$, defining $\mathit{sen}(X)$ to be the set of formulas generated from $a \in X$ and $Y \subseteq X$ according to

$$\varphi \;::=\; \top \mid \neg\varphi \mid \varphi \wedge \varphi' \mid \langle a \rangle \varphi \mid \Box_Y \varphi .$$

We interpret these formulas over A-states q, treating $\top$, $\neg$ and $\wedge$ as usual, and setting

$$q \models \langle a \rangle \varphi \quad \Longleftrightarrow \quad a \in q \text{ and } q_a \models \varphi$$

which generalizes to strings $s \in X^*$

$$q \models \langle s \rangle \varphi \quad \Longleftrightarrow \quad s \in q \text{ and } q_s \models \varphi$$

(recalling that $\langle a_1 \rangle \cdots \langle a_n \rangle \varphi$ is $\langle a_1 \rangle \cdots \langle a_n \rangle \varphi$). As for $\Box_Y$, we put

$$q \models \Box_Y \varphi \quad \Longleftrightarrow \quad (\forall s \in q \cap Y^*)\; q_s \models \varphi$$

relativizing $\Box$ to Y for reasons that will become clear below. When $\Box$ appears with no subscript, it is understood to carry the subscript A; i.e., $\Box$ is $\Box_{\mathsf{A}}$. Also, for $s \in \mathsf{A}^*$, we will often shorten the formula $\langle s \rangle \top$ to s, writing, for example, $\neg a$ for $\neg \langle a \rangle \top$.

4.1 Labels Refining Partitions

For any $\Sigma \in \mathit{Fin}(\mathsf{A})$ and Σ-state q, we can formulate the Myhill-Nerode equivalence $\approx_q$ between strings $s, s' \in \Sigma^*$ with the same extensions in q

$$s \approx_q s' \quad \Longleftrightarrow \quad (\forall w \in \Sigma^*)\; (sw \in q \iff s'w \in q)$$

(e.g., Hopcroft and Ullman 1979) in terms of derivatives

$$s \approx_q s' \quad \Longleftrightarrow \quad q_s = q_{s'}$$

from which it follows that q is regular iff its set

$$\{q_s \mid s \in \Sigma^*\}$$

of derivatives is finite. There are strict bounds on what we can discern with Σ and Σ-states. For example, the regular language

$$q' \;=\; \textsc{father}^* + \textsc{father}^* man$$

over $\Sigma' = \{\textsc{father}, man$ is a model of

$$man \;\wedge\; \Box(man \;\supset\; \langle \textsc{father} \rangle man)$$

(i.e., q' is a token of the record type man given by $\mathrm{eq}(man) = \{(\textsc{father}, man)\}$), with derivatives

$$q'_s \;=\; \begin{cases} q' & \text{if } s \in \textsc{father}^* \\ \{\epsilon\} & \text{if } s \in \textsc{father}^* man \\ \emptyset & \text{otherwise} \end{cases}$$

so that according to the definitions from the previous section (with δ given by derivatives of q'), the Σ'-state $\{\epsilon\}$ is null, and the label man is terminal. The Σ'-state q' does not differentiate between distinct tokens of man, although it is easy enough to introduce additional labels and formulas

$$\Box\,(man \;\supset\; (John \;\vee Peter \vee Other_{man:John,Peter})) \tag{8}$$

with

$$\Box\,(John \;\supset\; \neg\langle \textsc{father} \rangle John)$$

unless say, $John$ could apply to more than one particular, as suggested by

$$\Box\,(John \;\supset\; (John1 \;\vee John2 \vee Other_{John:1,2})).$$

Generalizing line (8) above, we can refine a label a to a finite partition from a set $\Sigma \in Fin(\mathsf{A})$, asserting

$$Partition_a(\Sigma) \;:=\; \Box\,(a \supset Uniq(\Sigma))$$

where $Uniq(\Sigma)$ picks out exactly one label in Σ

$$Uniq(\Sigma) \;:=\; \bigvee_{a \in \Sigma} (a \wedge \bigwedge_{a' \in \Sigma - \{a\}} \neg a')).$$

Co-occurrence restrictions on a set Σ of alternatives

$$Alt(\Sigma) \;:=\; \Box \bigwedge_{a \in \Sigma} \bigwedge_{a' \in \Sigma - \{a\}} \neg(a \wedge a')$$

(declaring any pair to be incompatible) is equivalent to the conjunction

$$\bigwedge_{a \in \Sigma} Partition_a(\Sigma).$$

And if the labels in Σ are understood to specify components, we might say a label marks a Σ-atom if it rules out a'-components for $a' \in \Sigma$

$$\begin{aligned} Atom_\Sigma(a) &:= \Box(a \supset \bigwedge_{a' \in \Sigma} \neg a') \\ &\iff Partition_a(\Sigma \cup \{a\}). \end{aligned}$$

That said, we arrived at $Partition_a(\Sigma)$ from *man* above through the example $\Sigma = \{John,\ Peter,\ Other_{man:John,Peter}\}$ of labels that differentiate between tokens of *man* rather than (as in the case of the record label FATHER or AGENT or THEME) specifying components. We can extend the example

$$Partition_{man}(\{John, Peter, Other_{man:John,Peter}\})$$

through a function $f : T \to Fin(\mathsf{A})$ from some finite set T of labels (representing types) such that for $a \in T$, $f(a)$ outlines a partition of a (just as $\{John,\ Peter,\ Other_{man:John,Peter}\}$ does for *man*). An *f-token* is then an A-state q such that

$$q \models \bigwedge_{a \in T} Partition_a(f(a))$$

making (as it were) a choice from $f(a)$, for each $a \in T$.

4.2 Reducts for Satisfaction

Given a string $s \in \mathsf{A}^*$ and a set $\Sigma \in Fin(\mathsf{A})$, the longest prefix of s that belongs to Σ^* is computed by the function $\pi_\Sigma : \mathsf{A}^* \to \Sigma^*$ defined by $\pi_\Sigma(\epsilon) := \epsilon$ and

$$\pi_\Sigma(as) := \begin{cases} a\,\pi_\Sigma(s) & \text{if } a \in \Sigma \\ \epsilon & \text{otherwise.} \end{cases}$$

The *Σ-reduct of* a language $q \subseteq \mathsf{A}^*$ is the image of q under π_Σ

$$q \restriction \Sigma := \{\pi_\Sigma(s) \mid s \in q\}\ .$$

If q is an A-state, then its Σ-reduct, $q \restriction \Sigma$, is a Σ-state and is just the intersection $q \cap \Sigma^*$ with Σ^*. Σ-reducts are interesting because satisfaction $\models$ of formulas in $sen(\Sigma)$ can be reduced to them.

Fact 4. *For every $\Sigma \in Fin(\mathsf{A})$, $\varphi \in sen(\Sigma)$ and A-state q,*

$$q \models \varphi \iff q \restriction \Sigma \models \varphi$$

and if, moreover, $s \in q \restriction \Sigma$, then

$$q \models \langle s \rangle \varphi \iff (q \restriction \Sigma)_s \models \varphi. \tag{9}$$

Fact 4 is proved by a routine induction on $\varphi \in sen(\Sigma)$. Were $sen(\Sigma)$ closed under $\Box = \Box_{\mathsf{A}}$, Fact 4 would fail for infinite A; hence, the relativized operators $\Box_Y$ for $Y \subseteq \Sigma$ in $sen(\Sigma)$.

There is structure lurking around Fact 4 that is most conveniently described in category-theoretic terms. For $\Sigma \in Fin(\mathsf{A})$, let $Q(\Sigma)$ be the category with

- Σ-states q as objects.
- pairs (q, s) such that $s \in q$ as morphisms from q to q_s, composing by concatenating strings

$$(q, s)\ ;\ (q_s, s')\ :=\ (q, ss')$$

with identities (q, ϵ).

To turn Q into a functor from $Fin(\mathsf{A})^{op}$ (with morphisms (Σ', Σ) such that $\Sigma \subseteq \Sigma' \in Fin(\mathsf{A})$) to the category **Cat** of small categories, we map a $Fin(\mathsf{A})^{op}$-morphism (Σ', Σ) to the functor $Q(\Sigma', \Sigma) : Q(\Sigma') \to Q(\Sigma)$ sending a Σ'-state q' to the Σ-state $q' \restriction \Sigma$, and the $Q(\Sigma')$-morphism (q', s') to the $Q(\Sigma)$-morphism $(q' \restriction \Sigma, \pi_\Sigma(s'))$. The *Grothendieck construction for* Q is the category $\int Q$ where

- objects are pairs (Σ, q) such that $\Sigma \in Fin(\mathsf{A})$ and q is a Σ-state.
- morphisms from (Σ', q') to (Σ, q) are pairs $((\Sigma', \Sigma), (q'', s))$ of $Fin(\mathsf{A})^{op}$-morphisms (Σ', Σ) and $Q(\Sigma)$-morphisms (q'', s) such that $q'' = q' \restriction \Sigma$ and $q = q''_s$.

(e.g., Tarlecki et al. 1991). $\int Q$ integrates the different categories $Q(\Sigma)$ (for $\Sigma \in Fin(\mathsf{A})$), lifting a $Q(\Sigma)$-morphism (q, s) to a $(\int Q)$-morphism from (Σ', q') to (Σ, q_s) whenever $\Sigma \subseteq \Sigma'$ and $q' \restriction \Sigma = q$.

Given a small category C, let us write $|C|$ for the set of objects of C. Thus, for $\Sigma \in Fin(\mathsf{A})$, $|Q(\Sigma)|$ is the set

$$|Q(\Sigma)|\ =\ \{q \subseteq \Sigma^*\ |\ q \neq \emptyset \text{ and } q \text{ is prefix}-\text{closed}\}$$

of Σ-states. Next, for $(\Sigma, q) \in |\int Q|$, let $Mod(\Sigma, q)$ be the full subcategory of $Q(\Sigma)$ with objects required to have q as a subset

$$|Mod(\Sigma, q)|\ :=\ \{q' \in |Q(\Sigma)|\ |\ q \subseteq q'\}.$$

That is, $|Mod(\Sigma, q)|$ is the set of Σ-states q' such that for all $s \in q$, $q' \models \langle s \rangle \top$. The intuition is that q is a form of record typing over Σ that allows us to simplify clauses such as

$$q' \models \langle s \rangle \varphi \quad \Longleftrightarrow \quad s \in q' \text{ and } {q'}_s \models \varphi \tag{10}$$

when $s \in q \subseteq q'$. The second conjunct in the righthand side of (10), $q'_s \models \varphi$, presupposes the first conjunct, $s \in q'$. We can lift that presupposition out of (10) by asserting that whenever $s \in q$ and $q \subseteq q'$,

$$q' \models \langle s \rangle \varphi \quad \Longleftrightarrow \quad q'_s \models \varphi.$$

This comes close to the equivalence (9) in Fact 4, except that Σ-reducts are missing. These reducts appear once we vary Σ, and step from $Q(\Sigma)$ to $\int Q$. Taking this step, we turn the categories $Mod(\Sigma, q)$ to a functor Mod from $\int Q$ to **Cat**, mapping a $\int Q$-morphism $\sigma = ((\Sigma', \Sigma), (q' \restriction \Sigma, s))$ from (Σ', q') to (Σ, q) to the functor

$$Mod(\sigma) : Mod(\Sigma', q') \to Mod(\Sigma, q)$$

sending $q'' \in |Mod(\Sigma', q')|$ to the s-derivative of its Σ-reduct, $(q'' \restriction \Sigma)_s$, and a $Mod(\Sigma', q')$-morphism (q'', s') to the $Mod(\Sigma, q)$-morphism $(q'' \restriction \Sigma, \pi_\Sigma(s'))$.

The syntactic counterpart of $Q(\Sigma)$ is $sen(\Sigma)$, which we turn into a functor sen matching Mod. A basic insight from Goguen and Burstall (1992) informing the present approach is the importance of a category **Sign** of *signatures* which the functor sen maps to the category **Set** of sets (and functions) and which Mod maps contravariantly to **Cat**. The definition of Mod above suggests that $\mathbf{Sign}^{op}$ is $\int Q$.[6] A $\int Q$-morphism from $\int Q$-objects (Σ', q') to (Σ, q) is determined uniquely by a string $s \in q' \restriction \Sigma$ such that

$$q = (q' \restriction \Sigma)_s \text{ and } \Sigma \subseteq \Sigma'. \tag{11}$$

Let $(\Sigma, q) \xrightarrow{s} (\Sigma', q')$ abbreviate the conjunction (11), which holds precisely if $((\Sigma', \Sigma), (q' \restriction \Sigma, s))$ is a $\int Q$-morphism from (Σ', q') to (Σ, q). Now for $(\Sigma, q) \in |\int Q|$, let

$$sen(\Sigma, q) \;=\; sen(\Sigma)$$

(ignoring q), and when $(\Sigma, q) \xrightarrow{s} (\Sigma', q')$, let

$$sen(\sigma) : sen(\Sigma) \to sen(\Sigma')$$

send $\varphi \in sen(\Sigma)$ to $\langle s \rangle \varphi \in sen(\Sigma')$. To see that an *institution* arises from restricting $\models$ to $|Mod(\Sigma, q)| \times sen(\Sigma)$, for $(\Sigma, q) \in |\int Q|$, it remains to check the *satisfaction condition*:

$$\text{whenever } (\Sigma, q) \xrightarrow{s} (\Sigma', q') \text{ and } q'' \in |Mod(\Sigma', q')| \text{ and } \varphi \in sen(\Sigma),$$

$$q'' \models \langle s \rangle \varphi \iff (q'' \restriction \Sigma)_s \models \varphi.$$

This follows from Fact 4 above, as s must be in $q' \restriction \Sigma$ and thus also in $q'' \restriction \Sigma$.

[6] That said, we might refine **Sign**, requiring of a signature (Σ, q) that q be a regular language. For this, it suffices to replace $\int Q$ by $\int R$ where $R : Fin(\mathsf{A})^{op} \to \mathbf{Cat}$ is the subfunctor of Q such that $R(\Sigma)$ is the full subcategory of $Q(\Sigma)$ with objects regular languages. We can make this refinement with*out* requiring that Σ-states in $Mod(\Sigma, q)$ be regular, forming $Mod(\Sigma, q)$ from Q (not R).

5 Conclusion

A process perspective is presented in Sect. 2 that positions frames to the left of a satisfaction predicate $\models$ for Hennessy-Milner logic over a set A of labels (or, from the perspective of Blackburn 1993, attributes). A is allowed to become arbitrarily large so that under identity of indiscernibles relative to A, a frame can be identified with a non-empty prefix-closed language over A. This identification is tried out in Sect. 3 on frames as types and particulars. A handle on A is provided by its finite subsets Σ, which are paired with languages $q \subseteq \Sigma^*$ for signatures (Σ, q), along which to reduce satisfaction to Σ-reducts and/or s-derivatives, for $s \in q$ (Fact 4). This plays out themes (mentioned in the introduction) of "semantics in flux" and "natural languages as collections of resources" (from Cooper and Ranta) in that, oversimplifying somewhat, s-derivatives specify transitions, while Σ-reducts pick out resources to use. The prominence of transitions (labeled by A) here contrasts strikingly with a finite-state approach to events (most recently described in Fernando 2015), where a single string (representing a timeline) appears to the left of a satisfaction predicate.[7] A Σ-state q to the left of $\models$ above offers a choice of strings, any number of which might be combined with other strings from other Σ'-states over different alphabets Σ'. A combination can be encoded as a string describing a timeline of resources used. This type/token distinction between languages and strings to the left of satisfaction has a twist; the languages are conceptually prior to the strings representing timelines, as nonexistent computer programs cannot run. Indeed, a profusion of alphabets Σ and Σ-states compete to make, in some form or other, a timeline that has itself a bounded signature (of a different kind). The processes through which a temporal realm is pieced together from bits of various frames call out for investigation.

While much remains to be done, let us be clear about what is offered above. A frame is structured according to strings of labels, allowing the set Σ of labels to vary over finite sets. That variation is tracked by a signature (Σ, q) picking out non-empty prefix-closed languages over Σ that contain the set q of strings over Σ. For example, Cooper's meaning function

$$\left(\lambda r : \begin{bmatrix} \textsc{agent} : all \\ \textsc{theme} : all \end{bmatrix}\right) \quad \begin{bmatrix} p_1 : smash(r) \\ p_2 : animate(r.\textsc{agent}) \\ p_3 : concrete(r.\textsc{theme}) \end{bmatrix}$$

is approximated by the signature (Σ, q) as the language L, where

$$\begin{aligned} \Sigma &= \{\textsc{agent},\ \textsc{theme},\ smash,\ animate,\ concrete\} \\ q &= \{\textsc{agent},\ \textsc{theme},\ \epsilon\} \\ L &= q \cup \{smash,\ \textsc{agent}\ animate,\ \textsc{theme}\ concrete\}. \end{aligned}$$

Further constraints can be imposed through formulas built with Boolean connectives and modal operators dependent on Σ — for instance, $Nom(\langle a \rangle \top)$. The

[7] This is formulated as an institution in Fernando (2014).

possibility of expanding Σ to a larger set makes the notion of identity as Σ-indiscernibility open-ended, and Σ a bounded but refinable level of granularity. A measure of satisfaction is taken in a finite-state calculus with, as Conway (1971) puts it, Taylor series

$$L \;=\; o(L) + \sum_{a \in \Sigma} aL_a$$

(from derivatives L_a), and a Grothendieck signature

$$\mathbf{Sign} \;=\; \int Q.$$

Acknowledgements. My thanks to Robin Cooper for discussions, to Glyn Morrill for help with presenting this paper at Formal Grammar 2015, and to two referees for comments and questions.

References

Barsalou, L.: Perceptual symbol systems. Behav. Brain Sci. **22**, 577–660 (1999)

Blackburn, P.: Modal logic and attribute value structures. In: de Rijke, M. (ed.) Diamonds and Defaults. Synthese Library, vol. 229, pp. 19–65. Springer, Netherlands (1993)

Brzozowski, J.: Derivatives of regular expressions. J. ACM **11**(4), 481–494 (1964)

Conway, J.H.: Regular Algebra and Finite Machines. Chapman and Hall, London (1971)

Cooper, R.: Type theory and semantics in flux. In: Philosophy of Linguistics, pp. 271–323. North-Holland (2012)

Cooper, R., Ranta, A.: Natural languages as collections of resources. In: Language in Flux: Dialogue Coordination, Language Variation, Change and Evolution, pp. 109–120. College Publications, London (2008)

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

Fernando, T.: Incremental semantic scales by strings. In: Proceedings of EACL 2014 Workshop on Type Theory and Natural Language Semantics, pp. 63–71. ACL (2014)

Fernando, T.: The semantics of tense and aspect: a finite-state perspective. In: Lappin, S., Fox, C. (eds.) The Handbook of Contemporary Semantic Theory, 2nd edn. Wiley, New York (2015)

Fillmore, C.: Frame semantics. In: Linguistics in the Morning Calm, pp. 111–137. Hanshin Publishing Co., Seoul (1982)

Goguen, J., Burstall, R.: Institutions: abstract model theory for specification and programming. J. ACM **39**(1), 95–146 (1992)

Hennessy, M., Milner, R.: Algebraic laws for non-determinism and concurrency. J. ACM **32**(1), 137–161 (1985)

Hopcroft, J., Ullman, J.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Reading (1979)

Kallmeyer, L., Osswald, R.: Syntax-driven semantic frame composition in lexicalized tree adjoining grammars. J. Lang. Model. **1**(2), 267–330 (2013)

Löbner, S.: Evidence for frames from human language. In: Gamerschlag, T., Gerland, D., Osswald, R., Petersen, W. (eds.) Frames and Concept Types. Studies in Linguistics and Philosophy, vol. 94, pp. 23–67. Springer, Switzerland (2014)

Muskens, R.: Data semantics and linguistic semantics. In: The Dynamic, Inquisitive, and Visionary Life of $\phi, ?\phi$, and $\Diamond\phi$: A Festschrift for Jeroen Groenendijk, Martin Stokhof, and Frank Veltman, pp. 175–183. Amsterdam (2013)

Osswald, R.: Semantics for attribute-value theories. In: Proceedings of Twelfth Amsterdam Colloquium, pp. 199–204. Amsterdam (1999)

Petersen, W., Osswald, T.: Concept composition in frames: focusing on genitive constructions. In: Gamerschlag, T., Gerland, D., Osswald, R., Petersen, W. (eds.) Frames and Concept Types. Studies in Linguistics and Philosophy, vol. 94, pp. 243–266. Springer, Switzerland (2014)

Rutten, J.J.M.M.: Automata and coinduction (an exercise in coalgebra). In: Sangiorgi, D., de Simone, R. (eds.) CONCUR 1998. LNCS, vol. 1466, pp. 194–218. Springer, Heidelberg (1998)

Tarlecki, A., Burstall, R., Goguen, J.: Some fundamental algebraic tools for the semantics of computation: Part 3. indexed categories. Theoret. Comput. Sci. **91**(2), 239–264 (1991)

Veltman, F.: Logics for conditionals. Ph.D. dissertation, University of Amsterdam (1985)