MSO with tests and reducts

Tim Fernando and **David Woods** and **Carl Vogel**
School of Computer Science and Statistics
Trinity College Dublin, Ireland
{tim.fernando,dwoods,carl.vogel}@tcd.ie

Abstract

Tests added to Kleene algebra (by Kozen and others) are considered within Monadic Second Order logic over strings, where they are likened to statives in natural language. Reducts are formed over tests and non-tests alike, specifying what is observable. Notions of temporal granularity are based on observable change, under the assumption that a finite set bounds what is observable (with the possibility of stretching such bounds by moving to a larger finite set). String projections at different granularities are conjoined by superpositions that provide another variant of concatenation for Booleans.

1 Introduction

Regular languages can be studied declaratively through formulas of Monadic Second-Order logic over strings (MSO; e.g., Libkin, 2010) or through equations built with the constructs $+, \cdot, ^*, 0, 1$ of a Kleene algebra (KA; e.g., Kozen, 1994). A KA with a subalgebra of *tests* forming a Boolean algebra is a KA with tests (KAT; e.g., Kozen, 1997). Tests are identified below with *statives* that serve as a basis for the approach to temporal semantics in linguistics initiated in Dowty (1979). This identification is justified by

(i) a guarded string interpretation of KAT (Kozen and Smith, 1996), in which tests form states, as conceived in Propositional Dynamic Logic (PDL, Fischer and Ladner, 1979), and

(ii) a notion of homogeneity associated (by Dowty and other linguists) with statives, and linked below to tests under a conception of time as observable change.

These two points are developed below in MSO using reducts. Kozen and Smith's definition of guarded strings is reformulated so that

(†) the MSO-sentence φ picking out guarded strings over actions Σ and tests B does *not* mention B (or their Boolean complements), asserting only that exactly one action occurs at every position except for the final one, where no action occurs.

Precisely what (†) means is taken up in section 2, with the help of reducts. Why (†) is significant becomes plain in section 3, where the reformulation is used to clarify the connection with tests and states in PDL.[1] A notion of temporal granularity based on observable change in MSO is built on projections that compress reducts. These projections are applied in section 4 to generalize interval networks from (Allen, 1983).

2 Guarded strings, MSO and reducts

For any finite set Σ, let Reg_Σ be the set of languages over the alphabet Σ accepted by finite automata. Then $\langle \mathsf{Reg}_\Sigma, \cup, \cdot, ^*, \emptyset, \epsilon\rangle$ is a KA — arguably, the Σ-canonical KA. For a KA with tests, we start in §2.1 with a finite set B of tests, and present the free Boolean algebra generated by B in terms of powersets 2^X of sets X. Strings over the alphabet $2^{B\cup\Sigma}$ are then used in §2.2 for an extension to a KA. This deviates tellingly from Kozen and Smith (1996)'s presentation of guarded strings over the alphabet $\Sigma \cup B \cup \overline{B}$ with Boolean complements $\overline{B}$ of B, reviewed in §2.3. The deviation is natural from the perspective of MSO, which is brought into the picture along with reducts in §2.4.

2.1 Finite free Boolean algebras

Given a set B, the set T_B of *Boolean terms over* B is the smallest set $\supseteq$-containing $B \cup \{0, 1\}$ that is closed under the binary connectives $+, \cdot$ and the unary connective c (for *c*omplements). Assuming

[1] We focus throughout on semantic intuitions relevant to our present purposes, leaving out details such as the equational axioms of KA or the precise language of PDL.

B is finite, the *free Boolean algebra generated by* B is

$$F(B) = \langle 2^{(2^B)}, \cup, \cap, \emptyset, 2^B, 2^B \setminus \cdot \rangle$$

(with addition $\cup$, multiplication $\cap$, and complement $2^B \setminus X$ of a subset X of 2^B). A B-*atom* is a subset q of B, and is used to interpret Boolean terms over B as follows

$$\begin{aligned} &[\![b]\!]_B := \{q \subseteq B \mid b \in q\} \ \text{ for } b \in B \\ &[\![0]\!]_B := \emptyset \qquad\qquad [\![1]\!]_B := 2^B \end{aligned}$$

and for terms $t, t' \in T_B$,

$$\begin{aligned} [\![t + t']\!]_B &:= [\![t]\!]_B \cup [\![t']\!]_B \\ [\![t \cdot t']\!]_B &:= [\![t]\!]_B \cap [\![t']\!]_B \\ [\![c(t)]\!]_B &:= [\![1]\!]_B \setminus [\![t]\!]_B. \end{aligned}$$

2.2 Guarded strings of sets

Next, given a set Σ disjoint from T_B, $\Sigma \cap T_B = \emptyset$, let the set $T_{\Sigma,B}$ of (Σ, B)-*terms* be the smallest set containing $\Sigma \cup T_B$ that is closed under the binary connectives $+, \cdot$ and the unary connective *. To extend the interpretation $[\![t]\!]_B$ of Boolean terms t over B to (Σ, B)-terms, we weaken the notion of a B-atom as follows. Let 2^B_Σ be the set

$$2^B_\Sigma \ := \ \{q \cup \{p\} \mid q \subseteq B \text{ and } p \in \Sigma\}$$

of sets obtained from a subset of B by adjoining an element of Σ. The set

$$\mathcal{G}^B_\Sigma \ := \ (2^B_\Sigma)^* 2^B \ \ (\subset (2^{B\cup\Sigma})^+$$

is generated by the rules

$$\frac{q \subseteq B}{q \in \mathcal{G}^B_\Sigma} \qquad \frac{q \subseteq B \quad p \in \Sigma \quad s \in \mathcal{G}^B_\Sigma}{(q \cup \{p\}) s \in \mathcal{G}^B_\Sigma}$$

to produce strings

$$(q_1 \cup \{p_1\}) \cdots (q_n \cup \{p_n\}) q_{n+1}$$

of length $n + 1$ (for $n \geq 0$), formed from $q_1 \cdots q_{n+1} \in (2^B)^{n+1}$ and $p_1 \cdots p_n \in \Sigma^n$. Let us call elements of $\mathcal{G}^B_\Sigma$ (Σ, B)-*guarded strings* (making B-atoms (Σ, B)-guarded strings of length 1). To interpret a (Σ, B)-term as a set of (Σ, B)-guarded strings, two bits of notation are handy.

(i) For any string s of length > 0, let α_s be the symbol that occurs first in s.

(ii) For any symbol q and language L, let $L[q]$ be the set of strings that, with q attached to the right, belong to L

$$L[q] \ := \ \{s \mid sq \in L\}.$$

Now, given sets L and L' of strings of length > 0, the Σ-*fused product of* L *and* L' is the set

$$L \bullet_\Sigma L' \ := \ \{ss' \mid s' \in L' \text{ and } s \in L[\alpha_{s'} \setminus \Sigma]\}$$

of strings ss' from $s' \in L'$ and s such that $sq \in L$ where q is $\alpha_{s'} \setminus \Sigma$. That is,

$$\begin{aligned} L \bullet_\Sigma L' \ = \ \{s \cdot_\Sigma s' \mid s \in L, \ s' \in L' \text{ and} \\ s \cdot_\Sigma s' \text{ is defined}\} \end{aligned}$$

where $\cdot_\Sigma$ is a partial binary function on strings of length > 0 such that

$$\begin{aligned} sq \cdot_\Sigma \alpha s' \text{ is defined } &\iff q = \alpha \setminus \Sigma \\ &\implies sq \cdot_\Sigma \alpha s' = s\alpha s'. \end{aligned}$$

Notice that if L and L' are both sets of B-atoms, then their Σ-fused product is just their intersection

$$L \bullet_\Sigma L' \ = \ L \cap L'.$$

Consequently, we can extend $[\![\cdot]\!]_B : T_B \to 2^{2^B}$ to an interpretation $[\![\cdot]\!]_{\Sigma,B} : T_{\Sigma,B} \to 2^{\mathcal{G}^B_\Sigma}$, setting

$$\begin{aligned} [\![t]\!]_{\Sigma,B} \ &:= \ [\![t]\!]_B \ \text{ for } t \in T_B \\ [\![p]\!]_{\Sigma,B} \ &:= \ \{(q \cup \{p\}) q' \mid q, q' \subseteq B\} \ \text{ for } p \in \Sigma \end{aligned}$$

and for all $t, t' \in T_{\Sigma,B}$,

$$\begin{aligned} [\![t + t']\!]_{\Sigma,B} \ &:= \ [\![t]\!]_{\Sigma,B} \cup [\![t']\!]_{\Sigma,B} \\ [\![t \cdot t']\!]_{\Sigma,B} \ &:= \ [\![t]\!]_{\Sigma,B} \bullet_\Sigma [\![t']\!]_{\Sigma,B} \\ [\![t^*]\!]_{\Sigma,B} \ &:= \ ([\![t]\!]_{\Sigma,B})^{\star_\Sigma} \end{aligned}$$

where the Σ-asterate $^{\star_\Sigma}$ is the Σ-fused analog of Kleene star

$$L^{\star_\Sigma} := \bigcup_{n \geq 0} L_n$$

with $L_0 := 2^B$ (the $\bullet_\Sigma$-identity for $2^{\mathcal{G}^B_\Sigma}$) and $L_{n+1} := L \bullet_\Sigma L_n$.

2.3 Strings in place of sets

Guarded strings in Kozen and Smith (1996) are conceived over an alphabet different from $2^{B\cup\Sigma}$ by fixing a string $b_1 \cdots b_n$ that enumerates

$$B = \{b_1, \ldots, b_n\}$$

	B-atom	alphabet	product
$\mathcal{G}^B_\Sigma$	$q \subseteq B$	$2^{\Sigma \cup B}$	$\bullet_\Sigma$
$\mathcal{G}_{\Sigma,B}$	$c_1 \cdots c_n \in \mathcal{A}_B$	$\Sigma \cup B \cup \overline{B}$	$\diamond_n$

Table 1: Guarded strings 2 ways, given Σ and B

without repetition (making n the cardinality of B). Each $b \in B$ is paired with a fresh test $\overline{b}$, relative to which a B-atom $q \subseteq B$ can be understood as n choices $c_1 \cdots c_n$ between b_i and $\overline{b_i}$, with

$$c_i := \begin{cases} b_i & \text{if } b_i \in q \\ \overline{b_i} & \text{otherwise.} \end{cases}$$

2^B is repackaged as the language

$$\mathcal{A}_B := (b_1 + \overline{b_1})(b_2 + \overline{b_2}) \cdots (b_n + \overline{b_n})$$

to turn $\mathcal{G}^B_\Sigma$ from §2.2 into the set

$$\mathcal{G}_{\Sigma,B} \; := \; (\mathcal{A}_B \Sigma)^* \mathcal{A}_B$$

of *guarded strings over* Σ *and* B, with alphabet

$$\Sigma \cup B \cup \overline{B} \quad \text{where} \quad \overline{B} \; := \; \{\overline{b_1}, \ldots, \overline{b_n}\}.$$

Every (Σ, B)-term t is then interpretable as a subset $[\![t]\!]$ of $\mathcal{G}_{\Sigma,B}$, with

$$[\![p]\!] = \{sps' \mid s, s' \in \mathcal{A}_B\} \qquad \text{for } p \in \Sigma$$

and for $b \in B$,

$$[\![b]\!] = \{s \in \mathcal{A}_B \mid s \in (B \cup \overline{B} - \{\overline{b}\})^+\}.$$

In place of the Σ-fused product $\bullet_\Sigma$, we have the *coalesced product* $\diamond_n$

$$L \diamond_n L' \; := \; \{s\hat{s}s' \mid s\hat{s} \in L, \; \hat{s}s' \in L' \text{ and } \mathrm{length}(\hat{s}) = n\}.$$

Inasmuch as the two KATs over $2^{\mathcal{G}^B_\Sigma}$ and $2^{\mathcal{G}_{\Sigma,B}}$ are isomorphic, it is tempting to dismiss the difference recorded in Table 1 as cosmetic. Nonetheless, there are reasons for preferring 2^B over $\mathcal{A}_B$ from the perspective of MSO, a natural home for Boolean tests, with or without atoms.

2.4 MSO and reducts

Given a finite set A, an *MSO$_A$-model* is understood (in this paper) to be a structure

$$\langle [n], S_n, \{U_a\}_{a \in A} \rangle$$

over the set $[n] := \{1, \ldots, n\}$ of integers from 1 to n (for some positive integer n), with the successor relation

$$S_n := \{(i, i+1) \mid i \in [n-1]\}$$

on $[n]$, and for each $a \in A$, a subset U_a of $[n]$. We can identify $\langle [n], S_n, \{U_a\}_{a \in A} \rangle$ with the string $\alpha_1 \cdots \alpha_n$ over the alphabet 2^A given by

$$\alpha_i \; := \; \{a \in A \mid i \in U_a\} \qquad \text{for } i \in [n]$$

making U_a the set of positions where a occurs

$$U_a \; = \; \{i \in [n] \mid a \in \alpha_i\}.$$

To construe a string $a_1 \cdots a_n \in A^+$ as an MSO$_A$-model, we lift it to $\boxed{a_1} \cdots \boxed{a_n} \in (2^A)^+$, drawing boxes instead of curly braces $\{,\}$ for sets *qua* string symbols, as opposed to sets *qua* languages.[2] Given a string s over the alphabet 2^A and a subset A' of A, the A'*-reduct of* s, $\rho_{A'}(s)$, is s intersected componentwise with A'

$$\rho_{A'}(\alpha_1 \cdots \alpha_n) \; := \; (\alpha_1 \cap A') \cdots (\alpha_n \cap A')$$

(Fernando, 2016). To illustrate, for $A = \Sigma \cup B$, the Σ-reduct of a string

$$(q_1 \cup \boxed{p_1}) \cdots (q_n \cup \boxed{p_n}) q_{n+1}$$

in $\mathcal{G}^B_\Sigma$ is

$$\boxed{p_1} \cdots \boxed{p_n}\boxed{}.$$

Indeed, we can describe $\mathcal{G}^B_\Sigma$ by embedding Σ into $2^{\Sigma \cup B}$ via

$$\Sigma_\Box \; := \; \{\boxed{p} \mid p \in \Sigma\}.$$

or by MSO$_A$-formulas built with unary predicate symbols P_a labeled by $a \in A$ and the binary predicate symbol S (for successors).

Proposition 1. *For any disjoint sets* Σ *and* B,

$$\begin{aligned} \mathcal{G}^B_\Sigma \; &= \; \{s \in (2^{B \cup \Sigma})^+ \mid \rho_\Sigma(s) \in {\Sigma_\Box}^* \Box\} \\ &= \; \{s \in (2^{B \cup \Sigma})^+ \mid s \models \forall x \chi_\Sigma(x)\} \end{aligned}$$

where $\chi_\Sigma(x)$ *is the MSO$_\Sigma(x)$-formula*

$$\exists y (xSy) \; \equiv \; \bigvee_{a \in \Sigma} P_a(x)$$

(saying x is non-final iff some $a \in \Sigma$ occurs at x)

[2] Although conflating a string s with the singleton language $\{s\}$ is usually harmless, it is dangerous to confuse, for instance, the empty language $\emptyset$ with the string $\Box$ (of length 1), or the language of two strings $\{a, a'\}$ with the single string $\boxed{a, a'}$.

conjoined with the MSO$_\Sigma(x)$-formula

$$\neg \bigvee_{a\in\Sigma} (P_a(x) \wedge \bigvee_{a'\in\Sigma\setminus\{a\}} P_{a'}(x))$$

(saying no two symbols from Σ occur at x).

Note that $\forall x\, \chi_\Sigma(x)$ is an MSO$_\Sigma$-sentence stating

($\dagger$) exactly one symbol from Σ occurs at every string position except for the last position, where no symbol from Σ occurs.

Inasmuch as ($\dagger$) describes a very particular encoding of guarded strings (applicable to $\mathcal{G}^B_\Sigma$ but not to $\mathcal{G}_{\Sigma,B}$), it is natural to ask: can we motivate ($\dagger$) without resorting to details of encoding? We will argue in section 3 that we can, observing for now that $\chi_\Sigma(x)$ makes no mention of B (belonging, as it does, to MSO$_\Sigma$).

The price for working with

$$\langle \mathcal{G}^B_\Sigma, \cup, \bullet_\Sigma, \emptyset, 2^B, 2^B \setminus \cdot\rangle$$

as opposed to Kozen and Smith (1996)'s KAT

$$\langle \mathcal{G}_{\Sigma,B}, \cup, \diamond_n, \emptyset, \mathcal{A}_B, \mathcal{A}_B \setminus \cdot\rangle$$

is a complication in the alphabet of strings interpreting MSO$_A$ from A to 2^A. But since MSO$_A$-models are already strings over 2^A, that price has already been paid. Rather it is the step from $\mathcal{G}^B_\Sigma$ to $\mathcal{G}_{\Sigma,B}$ that is costly, complicating the label set A with a set $\overline{B}$ of labels for complements of B. It is telling that a string in $\mathcal{G}_{\Sigma,B}$ satisfies the MSO$_{\{b,\overline{b}\}}$-biconditionals

$$P_{\overline{b}}(x) \equiv \neg P_b(x)$$

only at positions x where b or $\overline{b}$ occurs. By contrast, every string in $\mathcal{G}^B_\Sigma$ can be expanded to a MSO$_{\Sigma\cup B\cup\overline{B}}$-model satisfying

$$\forall x\ (P_{\overline{b}}(x) \equiv \neg P_b(x)) \qquad \text{for every } b \in B$$

(*not* that $\overline{b}$ is needed to interpret $T_{\Sigma,B}$ in $2^{\mathcal{G}^B_\Sigma}$).

A crude measure of the complexity of a regular language $L \subseteq (2^A)^+$ is given by

Proposition 2. *For any finite set A and regular language $L \subseteq (2^A)^+$, there is a smallest subset A' of A such that for some MSO$_{A'}$-formula φ,*

$$L = \{s \in (2^A)^+ \mid s \models \varphi\}.$$

Proposition 2 follows from

($\ddagger$) for all strings $s \in (2^A)^+$, subsets A' of A and MSO$_{A'}$-formulas φ,

$$s \models \varphi \iff \rho_{A'}(s) \models \varphi$$

and the fact that if A'' is another subset of A,

$$\rho_{A''}(\rho_{A'}(s)) = \rho_{A'\cap A''}(s).$$

Provable by induction on φ, ($\ddagger$) is an instance of the *satisfaction condition* characteristic of *institutions* (Goguen and Burstall, 1992), to which we shall return in §3.3 below.

If the least set A' that Proposition 2 associates with L is called the *grain of* L, then $\mathcal{G}^B_\Sigma$ has grain Σ (by Proposition 1 and a moment's reflection). Not so the regular language $\mathcal{G}_{\Sigma,B}$, whose image under the map

$$a_1 \cdots a_n \ \mapsto\ \boxed{a_1}\cdots\boxed{a_n}$$

has grain $\Sigma \cup B \cup \overline{B}$. Proposition 1 consigns B to the background (using MSO's propositional connectives to interpret the Boolean structure of a KAT), drawing all attention to Σ. Indeed, as conceived in PDL, tests belong in Σ — or so we argue in the next section (*pace* Kozen)

The remainder of this section fleshes out, for $A = \Sigma \cup B \cup \overline{B}$, an MSO$_A$-definition ψ^B_Σ of $\mathcal{G}_{\Sigma,B}$

$$\mathcal{G}_{\Sigma,B} \ =\ \{s \in A^+ \mid s \models \psi^B_\Sigma\}$$

and is best skipped by readers for whom $\chi_\Sigma(x)$ is ugly enough. We let ψ^B_Σ be $\forall x\ \psi_{\Sigma,B}(x)$ for $\psi_{\Sigma,B}(x)$ given with the help of some abbreviations. For $A' \subseteq A$, let one$_{A'}(x)$ be the MSO disjunction

$$\text{one}_{A'}(x) \ :=\ \bigvee_{a\in A'} P_a(x)$$

saying some symbol from A' occurs in position x, and let atm$_B(x_1 \ldots x_n)$ abbreviate

$$\bigwedge_{1\le i<n} x_i S x_{i+1} \wedge \bigwedge_{1\le i\le n} \text{one}_{\{b_i,\overline{b_i}\}}(x_i)$$

putting a string from $\mathcal{A}_B$ in $x_1 \ldots x_n$. Now, $\psi_{\Sigma,B}(x)$ is the conjunction of (1), (2) and (3) below, where (1) ensures $b_i + \overline{b_i}$ is followed by $b_{i+1} + \overline{b_{i+1}}$ for i from 1 to $n-1$

$$\bigwedge_{i=1}^{n-1} (\text{one}_{\{b_i,\overline{b_i}\}}(x) \supset \exists y(xSy \wedge$$
$$\text{one}_{\{b_{i+1},\overline{b_{i+1}}\}}(y))) \quad (1)$$

KAT	PDL
Boolean in B	formula φ
action in Σ	program (e.g., test $\varphi?$)
B-atom $\subseteq B$	state $\in Q$
guarded string	input/output pair $\in Q \times Q$

Table 2: KAT vs PDL

while (2) says $b_n + \overline{b_n}$ can only be followed by a symbol from Σ

$$\forall y(\text{one}_{\{b_n,\overline{b_n}\}}(x) \wedge xSy \supset \text{one}_\Sigma(y)) \qquad (2)$$

(allowing for the case where x is the last position of the string), and (3) puts atoms before and after x whenever a symbol from Σ occurs at x

$$\text{one}_\Sigma(x) \supset (\text{before}_B(x) \wedge \text{after}_B(x)) \qquad (3)$$

where $\text{before}_B(x)$ abbreviates

$$\exists x_1 \cdots \exists x_n \; (x_n Sx \wedge \text{atm}_B(x_1 \ldots x_n))$$

and $\text{after}_B(x)$ abbreviates

$$\exists x_1 \cdots \exists x_n \; (xSx_1 \wedge \text{atm}_B(x_1 \ldots x_n)).$$

3 Tests and observable change

A *test* in PDL is a program $\varphi?$ built from a proposition φ, where, given a set Q of states,

(i) φ is interpreted as the set $[\![\varphi]\!] \subseteq Q$ of states *satisfying* φ, and

(ii) a program p is interpreted as a binary relation $[\![p]\!]$ on Q consisting of pairs (q, q') such that

on input q, p can output q'

(iii) $\varphi?$ is a side-effect free test of φ that aborts on states that do not satisfy φ

$$[\![\varphi?]\!] \;:=\; \{(q, q) \mid q \in [\![\varphi]\!]\}.$$

A cursory comparison of PDL with KAT, summarised in Table 2, suggests KAT Booleans form PDL states (or B-atoms), raising the question:

> where is the KAT counterpart of $\varphi?$ in Σ, which is assumed disjoint from the set B of Booleans?

The present section fills this gap by introducing for every $b \in B$, a test $?b$ that is interpreted the way an action p in Σ is in KAT, albeit with more care than the "anything-goes" clause

$$[\![p]\!]_{\Sigma,B} := \{(q \cup \boxed{p})q' \mid q, q' \subseteq B\}$$

that accepts any input/output pair q, q'. To regulate the changes effected by an action in Σ, we introduce a labeled transition relation

$$E \;\subseteq\; 2^B \times \Sigma \times 2^B$$

and interpret each $p \in \Sigma$ as the subset

$$\{(q \cup \boxed{p})q' \mid E(q, p, q')\}$$

of $\mathcal{G}^B_\Sigma$ (writing $E(q, p, q')$ and $(q, p, q') \in E$ interchangably). The "anything-goes" interpretation is the special case

$$E = 2^B \times \Sigma \times 2^B.$$

But to capture the meaning of a test $?b$ in the manner PDL does for $\varphi?$, we require that

$$E(q, ?b, q') \implies b \in q \text{ and } q = q'$$

for all $q, q' \subseteq B$. To align the interpretation closer to the input/output semantics of PDL programs, we will interpret $[\![?b]\!]$ as

$$\{(q \cup \boxed{?b})q \mid q \subseteq B \text{ and } b \in q\}$$

and form B-reducts (removing actions $p \in \Sigma$ buried in guarded strings) before compressing them (according to bc from §3.1).

3.1 Regulated programs including tests

Given sets Σ and B, and for every $b \in B$, a label $?b \notin \Sigma \cup B$ such that

$$?b = ?b' \text{ only if } b = b',$$

let

$$\Sigma[B] := \Sigma \cup \{?b \mid b \in B\}.$$

We can then extend any set $E \subseteq 2^B \times \Sigma \times 2^B$ to

$$E_B \;:=\; E \cup \{(q, ?b, q) \mid q \subseteq B \text{ and } b \in q\}$$

and pick out the subset $\mathcal{G}_{\restriction E}$ (pronounced "G restricted by E") of $\mathcal{G}^{\Sigma[B]}_B$ generated by

$$\frac{q \subseteq B}{q \in \mathcal{G}_{\restriction E}} \qquad \frac{sq \in \mathcal{G}_{\restriction E} \qquad E_B(q, p, q')}{s(q \cup \boxed{p})q' \in \mathcal{G}_{\restriction E}}.$$

to interpret a term t from $T_{\Sigma[B],B}$ as a subset $[\![t]\!]_{\restriction E}$ of $\mathcal{G}_{\restriction E}$ by suitable adjustments to $[\![\cdot]\!]_{\Sigma,B}$. In particular, for $b \in B$,

$$[\![b]\!]_{\restriction E} \;=\; \{q \mid q \subseteq B \text{ and } b \in B\}$$

and for $p \in \Sigma[B]$,

$$[\![p]\!]_{\restriction E} \;=\; \{(q \cup \{p\})q' \mid q, q' \subseteq B \text{ and } E_B(q, p, q')\}$$

and for $\bullet_\Sigma$ as defined in §2.2,

$$[\![t \cdot t']\!]_{\restriction E} \;=\; [\![t]\!]_{\restriction E} \bullet_\Sigma [\![t']\!]_{\restriction E}.$$

Now, whereas

$$L \bullet_\Sigma L' = L \cap L' \text{ for } L, L' \subseteq 2^B,$$

the interpretation $[\![?b]\!]_{\restriction E}$ of a test $?b$ is not a subset of 2^B unless it is $\emptyset$.

To relate $[\![?b]\!]_{\restriction E}$ back to $[\![b]\!]_{\restriction E}$, a few definitions are helpful. Let us call a string $\alpha_1 \cdots \alpha_n$ *stutterless* if $\alpha_i \neq \alpha_{i+1}$ for all $i \in [n-1]$. The *block compression* $bc(s)$ *of* a string $s = \alpha_1 \cdots \alpha_n$ deletes from s every α_i such that $\alpha_i = \alpha_{i+1}$

$$bc(s) \;:=\; s \text{ if length}(s) < 2$$

$$bc(\alpha\alpha' s) \;:=\; \begin{cases} bc(\alpha' s) & \text{if } \alpha = \alpha' \\ \alpha\, bc(\alpha' s) & \text{otherwise.} \end{cases}$$

Clearly, $bc(s)$ is stutterless and

$$s \text{ is stutterless} \iff s = bc(s).$$

Moreover, if $?b$ were removed from the strings in $[\![?b]\!]_{\restriction E}$, then we would be left with strings qq such that $b \in q$, to which we can apply bc to get $[\![b]\!]_{\restriction E}$. We systematise the removal of elements of $\Sigma[B]$ from strings in $[\![t]\!]_{\restriction E}$ next, aligning our semantics with PDL's.

3.2 Observable change

For terms $t \in T_{\Sigma[B],B}$ and subsets C of $\Sigma[B] \cup B$, let us apply block compression bc to the C-reducts of strings in $[\![t]\!]_{\restriction E}$ for

$$[t]_{E,C} \;:=\; \{bc(\rho_C(s)) \mid s \in [\![t]\!]_{\restriction E}\}$$

and observe that for all $b \in B$,

$$[?b]_{E,B} \;=\; [b]_{E,B} \;=\; [\![b]\!]_{\restriction E}.$$

More generally, let us define a translation

$$\theta : T_{\Sigma[B],B} \to T_{\Sigma,B}$$

translating tests $?b$ back to b

$$\theta(?b) := b \qquad \text{for } b \in B$$

otherwise leaving t as is

$$\theta(a) := a \qquad \text{for } a \in \Sigma \cup B$$

$$\theta(t + t') := \theta(t) + \theta(t') \qquad \theta(t^*) := \theta(t)^*$$

$$\theta(t \cdot t') := \theta(t) \cdot \theta(t') \qquad \theta(c(t)) := c(\theta(t)).$$

Also, let us say Σ is *E-active* if for every $p \in \Sigma$,

$$E(q, p, q') \implies q \neq q'$$

for all $q, q' \subseteq B$ (requiring that states change under p).

Proposition 3. *For all $t \in T_{\Sigma[B],B}$,*

$$[t]_{E,B} \;=\; [\theta(t)]_{E,B}$$

and assuming Σ is E-active,

$$[\theta(t)]_{E,B} \;=\; \{\rho_B(s) \mid s \in [\![\theta(t)]\!]_{\restriction E}\}.$$

The two parts of Proposition 3 can be sharpened at the cost of complicating the notation.

Part 1 For all $t \in T_{\Sigma[B],B}$,

$$[t]_{E,C} = [\theta(t)]_{E,C}$$

for any set C disjoint from $\Sigma[B]$.

Given $p \in \Sigma$, let us say p is (E, C)*-observable* if

$$E(q, p, q') \implies q \cap C \neq q' \cap C$$

for all $q, q' \subseteq B$ (so that p is C-observably E-active).

Part 2 For all $t \in T_{\Sigma,B}$,

$$[t]_{E,C} = \{\rho_C(s) \mid s \in [\![t]\!]_{\restriction E}\}$$

assuming that every $p \in \Sigma$ from which t is formed is (E, C)-observable.

3.3 Actions for a specific Boolean

The condition that p is (E, C)-observable can be formulated in MSO$_{C \cup \{p\}}$ as

$$\forall x \forall y\; ((P_p(x) \wedge xSy) \supset \text{diff}_C(x, y)) \qquad (4)$$

where diff$_C(x, y)$ abbreviates the MSO-formula

$$\text{diff}_C(x, y) \;:=\; \bigvee_{b \in C} \neg(P_b(x) \equiv P_b(y))$$

saying x and y can be separated by a unary predicate with label from C. Dropping the action p from (4) results in the requirement that every temporal step S change C

$$\forall x \forall y\; (xSy \supset \text{diff}_C(x, y)) \qquad (\text{ntc}_C)$$

designated (ntc$_C$) for the slogan

$$\textit{no time without change}_C.$$

This slogan is behind the function bc_C that maps a string s to the block compression of its C-reduct

$$bc_C(s) := bc(\rho_C(s))$$

(turning $[\![t]\!]_{\restriction E}$ to $[\![t]\!]_{E,C}$ in §3.2).

Proposition 4. *For any $C \subseteq A$ and $s \in (2^A)^*$,*

$$s \models (ntc_C) \iff bc_C(s) = s$$

and

$$bc_C(bc_C(s)) = bc_C(s).$$

To understand the importance of the subscript C, recall that MSO satisfaction $\models$ has the property

(‡) for all strings $s \in (2^A)^+$, subsets C of A and MSO$_C$-sentences φ,

$$s \models \varphi \iff \rho_C(s) \models \varphi.$$

(‡) brings out a fundamental limitation of an MSO$_C$-sentence φ, its insensitivity to differences between strings with the same C-reduct.

The significance of the subscript C is easy to overlook when describing $\mathcal{G}_{\restriction E}$ in MSO. Consider from Proposition 1, the $\chi_\Sigma(x)$ conjunct

$$\neg\text{two}_\Sigma(x) := \neg \bigvee_{a\in\Sigma}(P_a(x) \wedge \bigvee_{a'\in\Sigma\setminus\{a\}} P_{a'}(x))$$

banning two programs in Σ from occurring simultaneously at x. The problem with running $p \in \Sigma$ simultaneously with $?b \notin \Sigma$ at x is that the state transitions they describe under E_B may clash. Indeed, programs in PDL and more generally, Dynamic Logic (Harel et al., 2000) are interpreted as executing in isolation; for instance, the PDL test $\varphi?$ ensures the input state does not change, and a random assignment $x :=?$ changes at most the value of x. In both cases, any change from a program running concurrently is ruled out. Put another way, $\chi_\Sigma(x)$'s conjunct $\neg\text{two}_\Sigma(x)$ expresses the assumption that each program in Σ is to be understood as covering all programs that might run at x.

By contrast, actions described in everyday speech are invariably partial in that

(i) their effects are bounded, and

(ii) they never occur in isolation.

Keeping (i) and (ii) in mind, and zeroing in on a specific Boolean $b \in B$, let us add labels $l(b)$ and $r(b)$ to Σ for actions that mark the *l*eft and *r*ight borders of b as follows. Let $\Delta_b^l(x)$ be the MSO$_{\{b\}}(x)$-formula

$$\Delta_b^l(x) := (\exists y)(xSy \wedge P_b(y)) \wedge \neg P_b(x)$$

putting x just before b becomes true, and let $\Delta_b^r(x)$ be the MSO$_{\{b\}}(x)$-formula

$$\Delta_b^r(x) := P_b(x) \wedge \neg(\exists y)(xSy \wedge P_b(y))$$

putting x at b's right border. We then use $\Delta_b^l(x)$ to define $P_{l(b)}$

$$\forall x\ (P_{l(b)}(x) \equiv \Delta_b^l(x)) \qquad (l_b)$$

and $\Delta_v^r(x)$ for $P_{r(v)}$

$$\forall x\ (P_{r(b)}(x) \equiv \Delta_b^r(x)) \qquad (r_b)$$

(Fernando, 2019). Now, replacing diff$_C(x, y)$ in (ntc$_C$) by

$$\text{border}_C(x) := \bigvee_{b\in C}(P_{l(b)}(x) \vee P_{r(b)}(x))$$

yields: no time without *borders*$_C$

$$\forall x\forall y\ (xSy \supset \text{border}_C(x)). \qquad (\text{ntb}_C)$$

More precisely,

$$(\bigwedge_{b\in C}(l_b) \wedge (r_b)) \supset ((\text{ntc}_C) \equiv (\text{ntb}_C))$$

since for every $b \in C$,

$$((l_b) \wedge (r_b) \wedge xSy) \supset$$
$$(P_b(x) \equiv P_b(y)) \equiv (P_{l(b)}(x) \vee P_{r(b)}(x))$$

suppressing $\forall x\forall y$ to simplify the notation. Returning now to points (i) and (ii) above, notice that under (l_b) and (r_b),

(i) the effects of $l(b)$ and $r(b)$ are confined to b and although

$$((l_b) \wedge (r_b)) \supset \neg\exists x(P_{l(b)}(x) \wedge P_{r(b)}(x))$$

means $l(b)$ cannot occur with $r(b)$,

(ii) $l(b)$ can occur with $l(b')$ or $r(b')$ for $b' \neq b$. [3]

Complex actions can be built from a finite set of b-specific actions $l(b)$ and $r(b)$, provided we stay away from the $\mathcal{G}_\Sigma^B$ postulate $\neg\text{two}_\Sigma(x)$, which effectively pretends actions are indivisible atoms.

[3] Approximating $l(b)$ by the Dynamic Logic program

$$(b = \text{false})?; b :=?; (b = \text{true})?$$

overshoots badly, having unbounded effects that go beyond $l(b)$ in banning any changes to b' different from b.

4 Projections and superpositions

Having re-interpreted concatenation $\cdot$ as $\bullet_\Sigma$ and $\diamond_n$ in section 2 so that its restriction to tests is Boolean conjunction, we present in this section yet another notion of conjunction for combining descriptions of change at varying granularities. We start with the descriptions in §4.1, computing their conjunctions in §4.2.

4.1 Some star-free descriptions

Given a subset C of some fixed set A (determining a fragment MSO_A) and a string s of subsets of C, let us agree the pair (C, s) describes the set of stutterless strings over the alphabet 2^A that bc_C maps to s.[4] That is, if we gather together all stutterless strings over 2^A in

$$\mathcal{L}_A := \{\mathit{bc}(s) \mid s \in (2^A)^*\}$$

then

$$[\![(C, s)]\!]_A \;:=\; \{s' \in \mathcal{L}_A \mid \mathit{bc}_C(s') = s\}.$$

To illustrate, for

$$s_1 = \boxed{}\boxed{1}\boxed{},$$

$[\![(\{1\}, s_1)]\!]_{\{1,2\}}$ consists of s_1, all strings from

$$\boxed{}(\boxed{2}\boxed{})^*\boxed{1}\boxed{}$$

and many more, including $\boxed{2}\boxed{2,1}\boxed{}\boxed{2}$. In general, for $s \in \mathcal{L}_A$, $[\![(A, s)]\!]_A$ is $\{s\}$. Otherwise, if C is a proper subset of A, then $[\![(C, s)]\!]_A$ is infinite. In either case, $[\![(C, s)]\!]_A$ is first-order definable with the transitive closure $<$ of S. That is, $[\![(C, s)]\!]_A$ is star-free.

Next, we interpret a finite subset $\mathcal{C}$ of $2^A \times \mathcal{L}_A$ as the intersection

$$[\![\mathcal{C}]\!]_A \;=\; \bigcap_{(C,s)\in\mathcal{C}} [\![(C, s)]\!]_A$$

of the interpretations of pairs (C, s) in $\mathcal{C}$. Notice $[\![\mathcal{C}]\!]_A$ is also star-free. Continuing the example above, if

$$\mathcal{C}_2 = \{(\{1\}, s_1), (\{2\}, s_2)\}$$

where $s_i = \boxed{}\boxed{i}\boxed{}$ then $[\![\mathcal{C}_2]\!]_{\{1,2\}}$ consists of exactly 13 strings, one for each of the interval relations from Allen (1983), such as

$$\boxed{}\boxed{2}\boxed{1,2}\boxed{2}\boxed{} \quad \text{depicting} \quad 1 \textit{ during } 2$$

(e.g., Fernando, 2016). Generalizing from 2 intervals to any integer $n \geq 2$, we can extend the set

$$\{(\{i\}, s_i) \mid i \in [n]\}$$

to a partial function $\mathcal{C}$ from $2^{[n]}$ to $\mathcal{L}_{[n]}$, defined on certain pairs $\{i, j\}$ which $\mathcal{C}$ maps to a string $\mathcal{C}(\{i, j\})$ depicting an Allen relation between i and j. The result is an interval network with node set $[n]$ and edge set

$$\{C \in \mathit{domain}(\mathcal{C}) \mid |C| = 2\},$$

each C in which is labeled by the Allen relation depicted by $\mathcal{C}(C)$. We can label the edge C by a set $L \subseteq \mathcal{L}_C$ if we loosen (C, s) to the pair (C, L), interpreted as the inverse image of L under bc_C restricted to $\mathcal{L}_A$

$$\begin{aligned}[\![(C, L)]\!]_A \;&:=\; \{s \in \mathcal{L}_A \mid \mathit{bc}_C(s) \in L\} \\ &=\; \bigcup_{s\in L} [\![(C, s)]\!]_A.\end{aligned}$$

For $A \subseteq A'$,

$$\begin{aligned}[\![\mathcal{C}]\!]_{A'} \;&=\; \{s \in \mathcal{L}_{A'} \mid \mathit{bc}_A(s) \in [\![\mathcal{C}]\!]_A\} \\ &=\; [\![(A, [\![\mathcal{C}]\!]_A)]\!]_{A'}\end{aligned}$$

since

$$\mathit{bc}_C(s) = \mathit{bc}_C(\mathit{bc}_{C'}(s)) \;\text{ when } C \subseteq C' \subseteq A.$$

Thus, we can calculate $[\![\mathcal{C}]\!]_A$ by concentrating on $\bigcup \mathit{domain}([\![\mathcal{C}]\!])$ before attending to the full set A

$$[\![\mathcal{C}]\!]_A \;=\; [\![(\hat{C}, [\![\mathcal{C}]\!]_{\hat{C}})]\!]_A \;\text{ for } \hat{C} := \bigcup \mathit{domain}(\mathcal{C}).$$

As §4.2 makes clear, $[\![\mathcal{C}]\!]_{\hat{C}}$ is always finite (unlike $[\![\mathcal{C}]\!]_A \supseteq [\![\mathcal{C}]\!]_{\hat{C}}$).

4.2 Conjunction as superposition

We now define, for any subsets C and C' of A, a binary operation $\&_{C,C'}$ on languages such that for all $s \in \mathcal{L}_C$ and $s' \in \mathcal{L}_{C'}$,

$$[\![\{(C, s), (C', s')\}]\!]_{C\cup C'} \;=\; \{s\} \,\&_{C,C'}\, \{s'\}$$

and more generally, for all $L \subseteq \mathcal{L}_C$ and $L' \subseteq \mathcal{L}_{C'}$,

$$L \,\&_{C,C'}\, L' \;=\; [\![(C, L)]\!]_{C\cup C'} \cap [\![(C', L')]\!]_{C\cup C'}.$$

As a first stab, observe that if $\&_\circ$ forms the componentwise union of strings of the same length

$$\alpha_1 \cdots \alpha_n \,\&_\circ\, \alpha'_1 \cdots \alpha'_n \;:=\; (\alpha_1 \cup \alpha'_1) \cdots (\alpha_n \cup \alpha'_n)$$

[4] The restriction here to stutterless strings is motivated by the Aristotelian dictum, *no time without change*, a C-relativization of which is enforced by bc_C (Proposition 4).

then

$$\rho_{C\cup C'}(s) \;=\; \rho_C(s)\;\&_\circ\;\rho_{C'}(s).$$

It will be useful to introduce rules (s0) and (s1)

$$\frac{}{\&(\epsilon,\epsilon,\epsilon)}\ \text{(s0)} \qquad \frac{\&(s,s',\hat{s})}{\&(\alpha s,\alpha' s,(\alpha\cup\alpha')\hat{s})}\ \text{(s1)}$$

that together generate $\&_\circ$ as the set of triples $(s,s',\hat{s})$ such that

$$\&(s,s',\hat{s}) \ \text{ is derivable from (s0) and (s1).}$$

To factor in bc for bc_C, we add the two rules (b1) and (b2)

$$\frac{\&(\alpha s,s',\hat{s})}{\&(\alpha s,\alpha' s',(\alpha\cup\alpha')\hat{s})}\ \text{(b1)}$$

$$\frac{\&(s,\alpha' s',\hat{s})}{\&(\alpha s,\alpha' s',(\alpha\cup\alpha')\hat{s})}\ \text{(b2)}$$

so that, for example, the language

$$[\![(\{1\},\boxed{}\boxed{1}\boxed{})]\!]_{\{1,2\}} \,\cap\, [\![(\{2\},\boxed{}\boxed{2}\boxed{})]\!]_{\{1,2\}}$$

of Allen relations between 1 and 2 (from §4.1) is the set of strings s such that

$$\&(\boxed{}\boxed{1}\boxed{},\boxed{}\boxed{2}\boxed{},s)$$

is derivable from (s0), (s1), (b1) and (b2). The intersection $[\![(C,s)]\!]_{C\cup C'} \cap [\![(C',s')]\!]_{C\cup C'}$ becomes trickier when $C\cap C' \neq \emptyset$ (as with the transitivity table in Allen (1983)). Accordingly, we refine the rules (s1), (b1) and (b2), adding the side conditions

$$\alpha\cap C' \subseteq \alpha' \quad\text{and}\quad \alpha'\cap C\subseteq\alpha$$

to these rules for

$$\frac{\&(s,s',\hat{s}) \quad \alpha\cap C'\subseteq\alpha' \quad \alpha'\cap C\subseteq\alpha}{\&(\alpha s,\alpha' s,(\alpha\cup\alpha')\hat{s})}\ \text{(s1)}_{C,C'}$$

and similarly for $\text{(b1)}_{C,C'}$ and $\text{(b2)}_{C,C'}$. Now, let $\&^C_{C'}(s,s',\hat{s})$ abbreviate:

$$\begin{gathered}\&(s,s',\hat{s}) \text{ is derivable from (s0),}\\ \text{(s1)}_{C,C'},\ \text{(b1)}_{C,C'} \text{ and } \text{(b2)}_{C,C'}.\end{gathered}$$

Then for all $s\in\mathcal{L}_C$, $s'\in\mathcal{L}_{C'}$ and $\hat{s}\in\mathcal{L}_{C\cup C'}$,

$$\&^C_{C'}(s,s',\hat{s}) \iff \hat{s}\in[\![\{(C,s),(C',s')\}]\!]_{C\cup C'}$$

and indeed, the definition we require is

$$L\ \&_{C,C'}\ L' := \{\hat{s}\in\mathcal{L}_{C\cup C'} \mid (\exists s\in L)(\exists s'\in L')\ \&^C_{C'}(s,s',\hat{s})\}$$

(Woods and Fernando, 2018).

4.3 More projections

Recalling the KAT dichotomy between Booleans in B and actions in Σ (paralleling that between formulas and programs in Dynamic Logic[5]) it should be noted that the sets C and C' have been construed throughout to be subsets of B. The MSO-formulas $\Delta^l_b(x)$ and $\Delta^r_b(x)$ introducing the actions $l(b)$ and $r(b)$ in §3.3 define a border translation from B to Σ under which bc becomes the removal $d_\Box$ of empty boxes underlying projections in the S-strings of Durand and Schwer (2008), with, for instance, the Allen relation 1 *during* 2 recast as

$$\boxed{l(2)}\boxed{l(1)}\boxed{r(1)}\boxed{r(2)}$$

(Fernando, 2019; Fernando and Vogel, 2019). This section has focused on bc (for tests/statives) to lighten the notation. We can adapt §§4.1, 4.2 for $C, C'\subseteq\Sigma$, putting $d_\Box$ in place of bc.

5 Conclusion

The present paper is essentially an argument for interpreting MSO$_A$ relative to strings over the alphabet 2^A, rather than strings over the alphabet A. The latter smuggles in an assumption $\forall x\ \mathsf{spec}_A(x)$ where $\mathsf{spec}_A(x)$ is the MSO$_A(x)$-formula

$$\bigvee_{a\in A}(P_a(x)\wedge\bigwedge_{a'\in A\setminus\{a\}}\neg P_{a'}(x))$$

specifying exactly one label from A for the string position x. For a KAT generated by Booleans B and actions Σ, the alphabet A may contain $B\cup\Sigma$ (not to mention $\overline{B}$), with the guarded string interpretation in (Kozen and Smith, 1996) imposing $\mathsf{spec}_B(x)$ and $\mathsf{spec}_\Sigma(x)$ at various positions x, treating states as Boolean atoms (absent in an infinite free Boolean algebra) and actions as programs running in isolation (as in Dynamic Logic). Neither $\mathsf{spec}_B(x)$ nor $\mathsf{spec}_\Sigma(x)$ is necessary or desirable for applications where descriptions of states and actions are partial. Section 2 challenges $\mathsf{spec}_B(x)$, slighting B with a Σ-reduct (Proposition 1), while section 3 puts notions of observable change (described in Propositions 3 and 4) ahead of $\mathsf{spec}_\Sigma(x)$ to account for tests. Casting spec aside, section 4 compresses C-reducts, for $C\subseteq B$, and conjoins them by superposition. (More in Fernando, To appear.)

[5]Linguistic papers applying Dynamic Logic to temporal semantics include Naumann (2001); Pustejovsky and Moszkowicz (2011).

Acknowledgments

Thanks to three anonymous referees for their comments, and to Heiko Vogler for encouragement. This research is supported by Science Foundation Ireland (SFI) through the CNGL Programme (Grant 12/CE/I2267) in the ADAPT Centre, `https://www.adaptcentre.ie`. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

References

J. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

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

I.A. Durand and S.R. Schwer. 2008. A tool for reasoning about qualitative temporal information: the theory of S-languages with a Lisp implementation. *Journal of Universal Computer Science*, 14(20):3282–3306.

T. Fernando. 2016. On regular languages over power sets. *Journal of Language Modelling*, 4(1):29–56.

T. Fernando. 2019. Projecting temporal properties, events and actions. In *Proceedings 13th International Conference on Computational Semantics*, pages 1–12. www.aclweb.org/anthology/W19-0401.

T. Fernando. To appear. Pictorial narratives and temporal refinement. In *Proc 29th Semantics and Linguistic Theory (SALT)*. UCLA/Cornell.

T. Fernando and C. Vogel. 2019. Prior probabilities of Allen interval relations over finite orders. In *Proc 11th International Conference on Agents and Artificial Intelligence (ICAART 2019), Special Session on Natural Language Processing in AI*. Prague, available in www.scss.tcd.ie/Tim.Fernando/NLPinAI_2019.pdf.

M.J. Fischer and R.E. Ladner. 1979. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211.

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

D. Harel, D. Kozen, and J. Tiuryn. 2000. *Dynamic Logic*. MIT Press.

D. Kozen. 1994. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390.

D. Kozen. 1997. Kleene algebra with tests. *Transactions on Programming Languages and Systems*, 19(3):427–443.

D. Kozen and F. Smith. 1996. Kleene algebra with tests: Completeness and decidability. In *Proc. 10th Int. Workshop Computer Science Logic (CSL'96)*, LNCS 1258, pages 244–259. Springer-Verlag.

L. Libkin. 2010. *Elements of Finite Model Theory*. Springer.

R. Naumann. 2001. Aspects of changes: a dynamic event semantics. *Journal of Semantics*, 18:27–81.

J. Pustejovsky and J.L. Moszkowicz. 2011. The qualitative spatial dynamics of motion in language. *Spatial Cognition and Computation*, 11:15–44.

D. Woods and T. Fernando. 2018. Improving string processing for temporal relations. *Proc. 14th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA-2018)*, pages 76–86.