Finite State Methods and Description Logics

Tim Fernando
Trinity College Dublin, Ireland
Tim.Fernando@tcd.ie

Abstract. The accepting runs of a finite automaton are represented as concepts in a Description Logic, for various systems of roles computed by finite-state transducers. The representation refines the perspective on regular languages provided by Monadic Second-Order Logic (MSO), under the Büchi-Elgot-Trakhtenbrot theorem. String symbols are structured as sets to succinctly express MSO-sentences, with auxiliary symbols conceived as variables bound by quantifiers.

1 Introduction

As declarative specifications of sets of strings accepted by finite automata (i.e. regular languages), regular expressions are far and away more popular than the formulas of *Monadic Second-Order Logic* (MSO), which, by a fundamental theorem due to Büchi, Elgot and Trakhtenbrot, pick out the regular languages (e.g. Thomas, 1997). Computational semantics, however, can hardly ignore MSO's model-theoretic perspective on strings with its computable notions of entailment. Furthermore, regular expressions lack the succinctness that MSO's Boolean connectives support. Both negation and conjunction blow up the size of regular expressions by an exponential or two (Gelade and Neven, 2008). A symptom of the problem is the exponential cost of mapping a finite automaton $\mathbb{A}$ to a regular expression denoting the language $\mathcal{L}(\mathbb{A})$ accepted by $\mathbb{A}$ (Ehrenfeucht and Zeiger, 1976; Holzer and Kutrib, 2010). A more economical declarative representation of $\mathcal{L}(\mathbb{A})$ is afforded by pairing a string $a_1a_2\cdots a_n$ in $\mathcal{L}(\mathbb{A})$ with a string $q_1q_2\cdots q_n$ of $\mathbb{A}$'s (internal) states q_i in the course of a run (by $\mathbb{A}$) accepting $a_1a_2\cdots a_n$. This representation involves expanding the alphabet of the strings, and subsequently contracting the alphabet. A simple way to carry this out is by forming strings $\alpha_1\alpha_2\cdots\alpha_n$ of sets α_i that we can, for instance, intersect with a fixed set B, defining a string homomorphism ρ_B for componentwise intersection with B

$$\rho_B(\alpha_1\cdots\alpha_n) \;:=\; (\alpha_1\cap B)\cdots(\alpha_n\cap B).$$

For example, assuming *no* state q_i belongs to the alphabet Σ of $\mathcal{L}(\mathbb{A})$,

$$\rho_\Sigma(\boxed{a_1, q_1}\cdots\boxed{a_n, q_n}) = \boxed{a_1}\cdots\boxed{a_n}$$

where we draw boxes instead of curly braces for sets used as string symbols.

The homomorphisms ρ_B are linked below to the treatment of variables in MSO. Given a finite alphabet A, MSO_A-sentences are formed from a binary relation symbol S (encoding successor) and a unary relation symbol U_a, for each $a \in A$. We then interpret an MSO_A-sentence against a string over the alphabet 2^A of subsets of A, deviating ever so slightly from the custom of interpreting against strings in A^+. Expanding the alphabet from A to 2^A accommodates a form of underspecification that, among other things, facilitates the interpretation of MSO_A-formulas relative to variable assignments. As for the homomorphisms ρ_B, the idea is that B picks out a subset of A, leaving each $a \in A$ that is *not* in B as an "auxiliary marker symbol" — a staple of finite-state language processing (Beesley and Karttunen, 2003; Yli-Jyrä and Koskenniemi, 2004; Hulden, 2009).

By focusing on the accepting runs of a finite automaton, the present paper strives to be relevant to finite-state language processing in general. But to understand its difference with say (Hulden, 2009), a few words about the language applications motivating it might be helpful. These applications concern not morphology, phonology, speech or even syntax but semantics — in particular, temporal semantics. The convenience of equating succession in a string with temporal succession is yet another

Proceedings of the 11th International Conference on Finite State Methods and Natural Language Processing, pages 63–71,
St Andrews–Sctotland, July 15–17, 2013.

reason to step from A up to 2^A (reading a boxed subset of A as a snapshot). It is a trivial enough matter to build a finite-state transducer between $(2^A)^*$ and $(A \cup \{[,]\})^*$ unwinding say, the string

$$\boxed{a, a'}\boxed{a'} \text{ of length 2 over the alphabet } 2^{\{a,a'\}}$$

to the string

$$[aa'][a'] \text{ of length 7 over the alphabet } \{a, a', [,]\}$$

and if we are to apply ρ_B, it is natural to choose the alphabet 2^A over $A \cup \{[,]\}$. There are, in any case, many more regular relations apart from ρ_B to consider for temporal semantics (Fernando, 2011), including those that change string length and the notion of temporal succession, i.e. granularity. A simple but powerful way of building a regular language from a regular relation R is by forming the inverse image of a regular language L under R, which we write $\langle R\rangle L$, following dynamic logic (e.g. Fischer and Ladner, 1979). The basic thrust of the present paper is to extend regular expressions by adding connectives for negation, conjunction and inverse images under certain regular relations.

This extension is dressed up below in *Description Logic* (DLs; Baader et al. 2003), with the languages $\mathcal{L}(\mathbb{A})$ accepted by automata $\mathbb{A}$ as DL-concepts (unary relations), and various regular relations including ρ_B (for different sets B) as DL-roles (binary relations). As in the *attributive language with complement* $\mathcal{ALC}$, DL-concepts C are closed under conjunction $C \wedge C'$, negation $\neg C$, and inverse images under DL-roles R, the usual DL notation for which, $(\exists R)C$, we replace by $\langle R\rangle C$ from dynamic logic, with

$$[\![\langle R\rangle C]\!] \;:=\; \{s \mid (\exists s' \in [\![C]\!])\; s[\![R]\!]s'\}.$$

Since DL-concepts and DL-roles in the present context, have clear intended interpretations, we will often drop the semantic brackets $[\![\cdot]\!]$, conflating an expression with its meaning.

MSO is linked to DL by a mapping of MSO-sentences φ to DL-concepts C_φ that reduces MSO-entailments $\models_{MSO}$ to concept inclusion

$$\varphi \models_{MSO} \psi \iff C_\varphi \subseteq C_\psi\,. \qquad (1)$$

Let us take care *not* to read (1) as stating MSO is *interpretable* in DL in the sense of (Tarski et al., 1953); no formal DL theory is mentioned in (1), only a particular (intended) interpretation over strings. What we vary is not the interpretation $[\![\cdot]\!]$ but the mapping $\varphi \mapsto C_\varphi$ establishing (1). Many different definitions of C_φ will do, and the main aim of the present work is to explore these possibilities, expressed as particular DL concepts and roles. In its account of regular languages, the Büchi-Elgot-Trakhtenbrot theorem says nothing about finite-state transducers, which are nonetheless instrumental in establishing the theorem. Behind the move to Description Logics is the view that the role of finite-state transducers in constructing regular languages merits scrutiny.

To dispel possible confusion, it is perhaps worth remarking that a relation computable by a finite-state transducer may have a transitive closure that no finite-state transducer can compute. A simple example is the function f that leaves all strings unchanged except those of the form $1^n0^{m+2}2^k$ which it maps to $1^{n+1}0^m2^{k+1}$. The intersection

$$1^*2^* \;\cap\; \{f^k(0^n) \mid k, n \geq 0\}$$

is the non-regular language $\{1^n2^n \mid n \geq 0\}$, precluding a finite-state transducer from computing the transitive closure of f. Commenting on the Description Logic counterpart $\mathcal{ALC}_{reg}$ to Propositional Dynamic Logic (Fischer and Ladner, 1979), Baader and Lutz write

> many of today's most used concept languages do not include the role constructors of $\mathcal{ALC}_{reg}$. The main reason is that applications demand an implementation of description logic reasoning, and the presence of the reflexive-transitive closure constructor makes obtaining efficient implementations much harder.

There should be no mistaking the interpretations of concepts and roles below for models of $\mathcal{ALC}_{reg}$. Most every role considered below (from ρ_B on) is, however, transitive and indeed can be viewed as some notion of "part of."

The remainder of this paper is organized as follows. The accepting runs of a finite automaton are expressed as a DL-concept in section 2. This is then used to present MSO's semantic set-up in section 3. A feature of that set-up is the expansion of MSO_A-models from A^+ to $(2^A)^+$, opening up possibilities of underspecification which section 4 explores alongside non-deterministic DL-roles (in addition to the inverse of ρ_B). Section

5 looks at more DL-roles that, unlike ρ_B, change length, and section 6 concludes, returning to auxiliary symbols, which are subject not only to ρ_B but the various regular relations formulated above as DL-roles.

2 Accepting runs as a DL-concept

One half of the aforementioned Büchi-Elgot-Trakhtenbrot theorem turns finite automata to MSO-sentences. This section adapts the idea behind that half in a DL setting, deferring details about MSO to the next section. An *accepting run of* a finite automaton $\mathbb{A}$ is a string $\boxed{a_1, q_1}\boxed{a_2, q_2}\cdots\boxed{a_n, q_n}$ such that

$$q_0 \overset{a_1}{\rightsquigarrow} q_1 \overset{a_2}{\rightsquigarrow} q_2 \overset{a_3}{\rightsquigarrow} \cdots \overset{a_n}{\rightsquigarrow} q_n \in F$$

where q_0 is $\mathbb{A}$'s initial state, $\rightsquigarrow$ is $\mathbb{A}$'s set of transitions (labeled by symbols), and F is $\mathbb{A}$'s set of final/accepting states. (Note $\mathbb{A}$ may be identified with such a triple $\langle \rightsquigarrow, q_0, F\rangle$.) Let us assume that $\mathbb{A}$'s set Q of states is disjoint from the alphabet Σ, and treat $\boxed{a_i, q_i}$ as a 2-element subset of $\Sigma \cup Q$, making an accepting run a string over the alphabet $2^{\Sigma\cup Q}$ of subsets of $\Sigma \cup Q$. Clearly, for any finite automaton $\mathbb{A}$, the set *AccRuns*$(\mathbb{A})$ of accepting runs of $\mathbb{A}$ is regular — simply modify $\mathbb{A}$'s transitions to

$$\{(q, \boxed{a, q'}, q') \mid q \overset{a}{\rightsquigarrow} q'\} .$$

The language $\mathcal{L}(\mathbb{A})$ accepted by $\mathbb{A}$ can then be formed applying ρ_Σ to $\mathbb{A}$'s accepting runs (setting aside the cosmetic difference between $\boxed{a}$ and a)

$$\begin{aligned}\mathcal{L}(\mathbb{A}) &\approx \{\rho_\Sigma(s) \mid s \in \textit{AccRuns}(\mathbb{A})\}\\ &= \langle {\rho_\Sigma}^{-1}\rangle\, \textit{AccRuns}(\mathbb{A}) .\end{aligned}$$

It remains to express the language *AccRuns*$(\mathbb{A})$ through conjunctions and negations of a small number of languages $\langle R\rangle L$ for certain (particularly simple) choices of R and L. One such R is componentwise inclusion $\trianglerighteq$ between strings of sets of the same length

$$\trianglerighteq \;:=\; \{(\alpha_1\cdots\alpha_n, \beta_1\cdots\beta_n) \mid \alpha_i \supseteq \beta_i \text{ for } 1 \le i \le n\}$$

(Fernando, 2004). For example, $s \trianglerighteq \rho_B(s)$ for all strings s of sets, and $\alpha \trianglerighteq \boxed{\;}$ for all sets α. Returning to *AccRuns*$(\mathbb{A})$, let us collect

(i) strings with final position containing an $\mathbb{A}$-final state in

$$L_F \;:=\; \langle \trianglerighteq\rangle \sum_{q\in F} \boxed{\;}^* \boxed{q}$$

(ii) strings whose first position contains a pair a, q such that $q_0 \overset{a}{\rightsquigarrow} q$ in

$$L_{q_0\rightsquigarrow} \;:=\; \langle \trianglerighteq\rangle \sum_{q_0 \overset{a}{\rightsquigarrow} q} \boxed{a, q}\boxed{\;}^*$$

and

(iii) (bad) strings containing $\boxed{q}\boxed{a, q'}$, for some triple $(q, a, q') \in Q \times \Sigma \times Q$ *out*side the set $\rightsquigarrow$ of transitions in

$$L_{\not\rightsquigarrow} \;:=\; \langle \trianglerighteq\rangle \sum_{q \not\overset{a}{\rightsquigarrow} q'} \boxed{\;}^* \boxed{q}\boxed{a, q'}\boxed{\;}^*$$

Also, for any set B, let *Spec*(B) be the set

$$\textit{Spec}(B) \;:=\; \langle \rho_B\rangle (\sum_{b\in B} \boxed{b})^*$$

of strings with exactly one element of B in each string position.

Proposition 1. *Let $\mathbb{A}$ be a finite automaton with transitions $\rightsquigarrow \subseteq Q \times \Sigma \times Q$, final states $F \subseteq Q$ and initial state q_0, The set AccRuns$(\mathbb{A}) - \{\epsilon\}$ of non-null accessible runs of $\mathbb{A}$ is*

$$\textit{Spec}(\Sigma) \cap \textit{Spec}(Q) \cap L_F \cap L_{q_0\rightsquigarrow} - L_{\not\rightsquigarrow}$$

intersected with the set $(2^{\Sigma\cup Q})^*$ of strings over the alphabet $2^{\Sigma\cup Q}$ of subsets of the finite set $\Sigma \cup Q$.

Given any finite set A, the restrictions to $(2^A)^*$ of the relations ρ_B and $\trianglerighteq$

$$\begin{aligned}\rho_B^A &:= \rho_B \cap ((2^A)^* \times (2^A)^*)\\ \trianglerighteq_A &:= \trianglerighteq \cap ((2^A)^* \times (2^A)^*)\end{aligned}$$

are computable by finite-state transducers. That is, the intersection mentioned in Proposition 1 with $(2^{\Sigma\cup Q})^*$ can be built into the relations ρ_Σ, ρ_Q and $\trianglerighteq$ for a characterization of *AccRuns*$(\mathbb{A})$ as well as $\mathcal{L}(\mathbb{A})$ within a description logic, all concepts in which are regular languages, and all roles in which are computable by finite-state transducers (keeping $\langle R\rangle L$ regular). The obvious question is how that characterization compares with MSO, to which we turn next.

3 MSO-models and reducts from ρ_B

Given a finite set A, let us agree that an *MSO_A-model* M is a tuple $\langle [n], S_n, \{[\![U_a]\!]\}_{a\in A}\rangle$ for some positive integer $n > 0$,[1] where

(i) $[n]$ is the set $\{1, 2 \ldots, n\}$ of positive integers from 1 to n

(ii) S_n is the relation encoding the successor/next relation on $[n]$

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

and for each $a \in A$,

(iii) $[\![U_a]\!]$ is a subset of $[n]$ interpreting the unary relation symbol U_a.

We can form MSO_A-models from strings over the alphabets A and (2^A) as follows. Given a string $s = a_1a_2\cdots a_n \in A^n$, let $\mathrm{Mod}(s)$ be the MSO_A-model $\langle [n], S_n, \{[\![U_a]\!]\}_{a\in A}\rangle$ interpreting U_a as the set of string positions i occupied by a

$$[\![U_a]\!] \;=\; \{i \in [n] \mid a_i = a\}$$

for each $a \in A$. Expanding the alphabet A to 2^A, a string $\alpha_1\alpha_2\cdots\alpha_n \in (2^A)^n$ induces the MSO_A-model $\langle [n], S_n, \{[\![U_a]\!]\}_{a\in A}\rangle$ interpreting U_a as the set of positions i filled by boxes containing a

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

for each $a \in A$. Conversely, given an MSO_A-model $M = \langle [n], S_n, \{[\![U_a]\!]\}_{a\in A}\rangle$, let $\mathit{str}(M)$ be the string $\alpha_1\cdots\alpha_n \in (2^A)^n$ where

$$\alpha_i \;:=\; \{a \in A \mid i \in [\![U_a]\!]\}\,.$$

That is, for all $a \in A$ and $i \in [n]$,

$$a \in \alpha_i \iff i \in [\![U_a]\!]$$

making the map $M \mapsto \mathit{str}(M)$ a bijection between MSO_A-models and $(2^A)^+$.

Proposition 2. *Given an MSO_A-model M, the following are equivalent.*

(i) M is $\mathrm{Mod}(s)$ for some $s \in A^+$

(ii) M satisfies the MSO_A-sentence

$$(\forall x) \bigvee_{a\in A} (U_a(x) \wedge \bigwedge_{a'\in A-\{a\}} \neg U_{a'}(x))$$

(saying the non-empty $[\![U_a]\!]$'s partition the universe)

(iii) $\mathit{str}(M) \in \mathit{Spec}(A)$

(iv) $\mathit{str}(M) = \boxed{a_1}\cdots\boxed{a_n}$ for some $a_1\cdots a_n \in A^+$.

While the Büchi-Elgot-Trakhtenbrot theorem (BET) concerns MSO_A-models $\mathrm{Mod}(s)$ given by strings $s \in A^+$, the wider class of MSO_A-models (isomorphic to $(2^A)^+$) is useful for encoding variable assignments crucial to the second half of BET, asserting the regularity of MSO-sentences. More precisely, the remainder of this section is devoted to defining regular languages $\mathcal{L}_A(\varphi)$ for MSO_A-sentences φ establishing

Proposition 3. *For every MSO_A-sentence φ, there is a regular language $\mathcal{L}_A(\varphi)$ such that for every MSO_A-model M,*

$$M \models \varphi \iff \mathit{str}(M) \in \mathcal{L}_A(\varphi).$$

Insofar as the models $\mathrm{Mod}(s)$ given by strings $s \in A^+$ differ from those given by strings over (2^A) via str^{-1}, Proposition 3 differs from the half of BET saying MSO-sentences define regular languages. There is no denying, however, that the difference is slight.

Be that as it may, I claim the expansion of A to 2^A leads to a worthwhile simplification, as can be seen by proving Proposition 3 as follows. Given a positive integer n, an *n-variable assignment* f is a function whose domain is a finite set $\mathit{Var} = \mathit{Var}_1 \cup \mathit{Var}_2$ of first-order variables $x \in \mathit{Var}_1$ that f maps to an integer $f(x) \in [n]$ and second-order variables $X \in \mathit{Var}_2$ that f maps to a set $f(X) \subseteq [n]$. Then if M is a MSO_A-model over $[n]$,

$$M, f \models U_a(x) \iff f(x) \in [\![U_a]\!]$$

and

$$M, f \models X(x) \iff f(x) \in f(X).$$

[1] There is, of course, a rich theory of infinite MSO-models, given by infinite strings (e.g. Thomas, 1997). We focus here on finite models/strings, which suffice for many applications; more in section 6 below.

We can package the pair M, f as the $\mathrm{MSO}_{A \cup Var}$-model M^f over $[n]$ identical to M on U_a's for $a \in A$, with interpretations

$$\begin{aligned} [\![U_X]\!] &= f(X) \qquad \text{for } X \in \mathit{Var}_2 \\ [\![U_x]\!] &= \{f(x)\} \qquad \text{for } x \in \mathit{Var}_1. \end{aligned}$$

Note $[\![U_x]\!]$ intersects $[\![U_X]\!]$ if $M, f \models X(x)$, which is to say $X(x)$ entails the negation of $\mathit{spec}(\{X, x\})$, where $\mathit{spec}(A)$ is the MSO_A-sentence that Proposition 2 mentions in (ii)

$$(\forall x) \bigvee_{a \in A} (U_a(x) \wedge \bigwedge_{a' \in A - \{a\}} \neg U_{a'}(x)).$$

In other words, to treat a pair M, f as an $\mathrm{MSO}_{A \cup Var}$-model (and an MSO_A-formula φ with free variables in *Var* as an $\mathrm{MSO}_{A \cup Var}$-sentence), the step from A to 2^A is essential. Turning model expansions around, given $B \subseteq A$, we define the *B-reduct of* an MSO_A-model M to be the MSO_B-model M_B obtained from M after throwing out all interpretations $[\![U_a]\!]$ for $a \in A - B$. It is easy to see that the homomorphisms ρ_B yield B-reducts:

$$\mathit{str}(M_B) = \rho_B(\mathit{str}(M))$$

(for all MSO_A-models M). Proceeding now to the languages $\mathcal{L}_A(\varphi)$ in Proposition 3, observe that we can picture $U_a(x)$ as the language

$$L(a, x) \;:=\; (\boxed{\;} + \boxed{a})^* \boxed{a, x} (\boxed{\;} + \boxed{a})^*$$

inasmuch as

$$M, f \models U_a(x) \iff \rho^{A \cup Var}_{\{a,x\}}(\mathit{str}(M^f)) \in L(a, x).$$

For the remainder of this section, let $A' \subseteq A \cup \mathit{Var}$. We put, for $a, x \in A'$,

$$\mathcal{L}_{A'}(U_a(x)) \;:=\; \langle \rho^{A'}_{\{a,x\}} \rangle \, L(a, x).$$

Similarly, for $X, x \in A'$, let $\mathcal{L}_{A'}(X(x))$ be

$$\langle \rho^{A'}_{\{X,x\}} \rangle \, (\boxed{\;} + \boxed{X})^* \boxed{X, x} (\boxed{\;} + \boxed{X})^*$$

and for $x, y \in A'$,

$$\begin{aligned} \mathcal{L}_{A'}(x = y) \;&:=\; \langle \rho^{A'}_{\{x,y\}} \rangle \, \boxed{\;}^* \boxed{x, y} \boxed{\;}^* \\ \mathcal{L}_{A'}(S(x, y)) \;&:=\; \langle \rho^{A'}_{\{x,y\}} \rangle \, \boxed{\;}^* \boxed{x} \boxed{y} \boxed{\;}^*. \end{aligned}$$

We also put

$$\begin{aligned} \mathcal{L}_{A'}(\varphi \wedge \psi) \;&:=\; \mathcal{L}_{A'}(\varphi) \cap \mathcal{L}_{A'}(\psi) \\ \mathcal{L}_{A'}(\neg \varphi) \;&:=\; (2^{A'})^+ - \mathcal{L}_{A'}(\varphi). \end{aligned}$$

As for quantification, for $v \in \mathit{Var}$, let $\sigma^{A'}_v$ be the inverse of $\rho^{A' \cup \{v\}}_{A'}$,[2] and set

$$\begin{aligned} \mathcal{L}_{A'}(\exists X.\varphi) \;&:=\; \langle \sigma^{A'}_X \rangle \, \mathcal{L}_{A' \cup \{X\}}(\varphi) \\ \mathcal{L}_{A'}(\exists x.\varphi) \;&:=\; \langle \sigma^{A'}_x \rangle \, (\mathcal{L}_{A' \cup \{x\}}(\varphi) \cap \\ &\qquad \mathcal{L}_{A' \cup \{x\}}(x = x)) \end{aligned}$$

where the intersection with $\mathcal{L}_{A' \cup \{x\}}(x = x)$ insures that x is treated as a first-order variable in φ (occurring in exactly one string position).

Taking stock, to define the regular language $\mathcal{L}_A(\varphi)$ required by Proposition 3 for an MSO_A-sentence φ with variables from a finite set *Var* of variables, we form $\mathcal{ALC}$-concepts from

(i) the primitive concepts

$$\begin{aligned} &L(a, x),\ L(X, x), \\ &\boxed{\;}^* \boxed{x, y} \boxed{\;}^*,\ \boxed{\;}^* \boxed{x} \boxed{y} \boxed{\;}^*,\ (2^B)^+ \end{aligned}$$

for $a \in A$ and $x, X, y \in \mathit{Var}$ and $B \subseteq A \cup \mathit{Var}$, and

(ii) the roles

$$\rho^{A'}_B,\ \sigma^{A'}_v$$

for $B \subseteq A' \subseteq A \cup \mathit{Var}$ and $v \in \mathit{Var}$.

4 B-specified strings and containment $\sqsupseteq$

Recall that $\mathit{spec}(B)$ is the MSO_B-sentence saying every string position has exactly one symbol from B

$$(\forall x) \bigvee_{b \in B} (U_b(x) \wedge \bigwedge_{b' \in B - \{b\}} \neg U_{b'}(x))$$

and observe that there is no trace of σ^A_x or complementation or intersection (interpreting $\neg \exists x$ and $\wedge$) in the language

$$\mathit{Spec}_A(B) \;:=\; \langle \rho^A_B \rangle (\sum_{b \in B} \boxed{b})^*$$

of strings encoding MSO_A-models satisfying $\mathit{spec}(B)$, for $B \subseteq A$. That is, although $\mathit{Spec}_A(B)$ and $\mathcal{L}_A(\mathit{spec}(B))$ from the previous section specify the same set of strings, they differ as expressions, suggesting different automata. Section 2 provides yet more expressions for automata, drawing on a different pool of primitive concepts and roles.

[2] We could instead move the inverse out of the relation R, allowing $\langle R \rangle$ to go not only to the left of L as in $\langle R \rangle L$ but also to its right for the image $L\langle R \rangle$ of L under R.

In addition to componentwise inclusion $\trianglerighteq$ (a non-deterministic generalization of the functions ρ_B that dispenses with the subscript B), it is useful to define relations between strings of different lengths, including those that pick out prefixes

$$\mathit{prefix}_A \;:=\; \{(ss', s) \mid s, s' \in (2^A)^*\}$$

and suffixes

$$\mathit{suffix}_A \;:=\; \{(ss', s') \mid s, s' \in (2^A)^*\}.$$

Leaving out the subscripts $A = \Sigma \cup Q$ for notational simplicity, we can describe three of the languages in Proposition 1 (section 2) as

$$\begin{aligned} L_F \;&=\; \langle\trianglerighteq\rangle\langle \mathit{suffix}\rangle \sum_{q\in F} \boxed{q} \\ L_{q_0\rightsquigarrow} \;&=\; \langle\trianglerighteq\rangle\langle \mathit{prefix}\rangle \sum_{q_0 \overset{a}{\rightsquigarrow} q} \boxed{a, q} \\ L_{\not\rightsquigarrow} \;&=\; \langle\trianglerighteq\rangle\langle \mathit{suffix}\rangle\langle \mathit{prefix}\rangle \sum_{q \not\overset{a}{\rightsquigarrow} q'} \boxed{q}\boxed{a, q'} \end{aligned}$$

zeroing in on the substrings of length ≤ 2 that are of interest. It is convenient to abbreviate $\langle\trianglerighteq\rangle\langle \mathit{suffix}\rangle\langle \mathit{prefix}\rangle L$ (equivalently, $\langle\trianglerighteq\rangle\ \boxed{\ }^* L \boxed{\ }^*$) to $\langle\sqsupseteq\rangle L$, effectively defining *containment* $\sqsupseteq$ to be the relational composition of componentwise inclusion $\trianglerighteq$ with *suffix* and *prefix*

$$\sqsupseteq \;:=\; \trianglerighteq \,;\, \mathit{suffix} \,;\, \mathit{prefix}$$

(and $\sqsupseteq_A$ as $\trianglerighteq_A; \mathit{suffix}_A; \mathit{prefix}_A$). Writing $\mathcal{E}_A(x)$ for the set

$$\mathcal{E}_A(x) \;:=\; \mathcal{L}_A(x = x)$$

of strings in $(2^A)^+$ in which x occurs exactly once, we have

$$\begin{aligned} \mathcal{L}_A(U_a(x)) \;&=\; \mathcal{E}_A(x) \cap \langle\sqsupseteq\rangle \boxed{a, x} \\ \mathcal{L}_A(X(x)) \;&=\; \mathcal{E}_A(x) \cap \langle\sqsupseteq\rangle \boxed{X, x} \\ \mathcal{L}_A(x = y) \;&=\; \mathcal{E}_A(x) \cap \mathcal{E}_A(y) \;\cap\; \langle\sqsupseteq\rangle \boxed{x, y} \\ \mathcal{L}_A(S(x, y)) \;&=\; \mathcal{E}_A(x) \cap \mathcal{E}_A(y) \;\cap\; \langle\sqsupseteq\rangle \boxed{x}\boxed{y} \end{aligned}$$

and apart from $\mathcal{E}_A(x)$, primitive concepts given by strings of length ≤ 2 will do. The locality (in such short strings) is obscured in our ρ^A_B-based analysis of $\mathcal{L}_A(\varphi)$ in the previous section, under which

$$\begin{aligned} \mathcal{E}_A(x) \;&=\; \langle \rho^A_{\{x\}}\rangle \boxed{\ }^* \boxed{x} \boxed{\ }^* \\ &=\; \langle\sqsupseteq_A\rangle \boxed{x} \;-\; \langle\sqsupseteq\rangle \boxed{x}\boxed{\ }^*\boxed{x}. \end{aligned} \tag{2}$$

An existence predicate of sorts, $\mathcal{E}_A(x)$ is presuppositional in the same way for MSO that $\mathit{Spec}_A(B)$ is for the accepting runs of finite automata; $\mathcal{E}_A(x)$ and $\mathit{Spec}_A(B)$ are general, non-local background requirements imposed indiscriminately on models and automata, in contrast to assertions (in the foreground) that focus on short substrings, picking out specific models and automata.

An instructive test case is provided by the transitive closure $<$ of S; an $\mathrm{MSO}_{\{x,y\}}$-sentence saying that $x < y$ is

$$\begin{aligned} \exists X\ ((\forall u, v)(X(u) \wedge S(u, v) \supset X(v)) \\ \wedge\, X(y) \wedge \neg X(x)). \end{aligned}$$

Second-order quantification aside, the obvious picture to associate with $x < y$ is $\boxed{x}\boxed{\ }^*\boxed{y}$, which is built into the representation

$$\begin{aligned} \mathcal{L}_A(x < y) \;&=\; \mathcal{E}_A(x) \cap \mathcal{E}_A(y) \cap \\ &\qquad \langle\sqsupseteq\rangle \boxed{x}\boxed{\ }^*\boxed{y} \end{aligned} \tag{3}$$

of MSO_A-models satisfying $x < y$ (with $x, y \in A$). In both lines (2) and (3) above, arbitrarily long substrings from $\boxed{\ }^*$ occur, that we will show how to compress next.

5 Intervals and compression

Given $e \in A$, we can state that the set $[\![U_e]\!]$ represented by a symbol $e \in A$ is an interval through the $\mathrm{MSO}_{\{e\}}$-sentence

$$\exists x\ U_e(x) \wedge \neg\exists y\ \mathrm{gap}_e(y) \tag{4}$$

where $\mathrm{gap}_e(y)$ abbreviates the $\mathrm{MSO}_{\{e,y\}}$-sentence

$$\neg U_e(y) \wedge \exists u \exists v\ (u < y \wedge y < v \wedge U_e(u) \wedge\ U_e(v)).$$

We can translate (4) into a regular language, applying the recipes above. But a more concise and perspicuous representation is provided by defining a function bc that compresses a string s as follows. Let $\mathit{bc}(s)$ compress blocks β^n of $n > 1$ consecutive occurrences in s of the same symbol β to a single β, leaving s otherwise unchanged

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

For example,

$$\mathit{bc}(\boxed{\ }\boxed{e}\boxed{e}\boxed{e, y}\boxed{e}) \;=\; \boxed{\ }\boxed{e}\boxed{e, y}\boxed{e}$$

and in general, bc outputs only stutter-free strings, where a string $\beta_1\beta_2\cdots\beta_n$ is *stutter-free* if $\beta_i \neq \beta_{i+1}$ for i from 1 to $n-1$. Observe that $[\![U_e]\!]$ is an interval precisely if

$$bc(\rho_{\{e\}}(str(M))) \in (\boxed{\ } + \epsilon)\boxed{e}(\boxed{\ } + \epsilon).$$

Compressing further, we can delete initial and final empty boxes through *unpad*

$$unpad(s) \;:=\; \begin{cases} unpad(s') & \text{if } s = \boxed{\ }s' \text{ or} \\ & \text{else } s = s'\boxed{\ } \\ s & \text{otherwise} \end{cases}$$

and collect all strings in $(2^A)^+$ representing MSO_A-models in which e is an interval in

$$Interval_A(e) \;:=\; \langle\rho^A_{\{e\}}\rangle\langle bc\rangle\langle unpad\rangle\,\boxed{e}.$$

Defining π^A_B to be the composition of ρ^A_B with bc and *unpad*

$$\pi^A_B(s) \;:=\; unpad(bc(\rho^A_B(s)))$$

we have

$$Interval_A(e) \;=\; \langle\pi^A_{\{e\}}\rangle\boxed{e}$$

and, as promised at the end of section 4, we can eliminate $\boxed{\ }^*$ from (2)

$$\mathcal{E}_A(x) \;=\; \langle\pi^A_{\{x\}}\rangle\boxed{x} \;-\; \langle\sqsupseteq\rangle\boxed{x}\boxed{x}$$

and from (3)

$$\mathcal{L}_A(x<y) \;=\; \mathcal{E}_A(x) \cap \mathcal{E}_A(y) \cap \langle\pi_{\{x,y\}}\rangle\,(\boxed{x}\boxed{y} + \boxed{x}\boxed{\ }\boxed{y})$$

(dropping the superscript A on π^A_B when possible). Stepping from one interval e to a finite set E of such, let

$$\begin{aligned} Interval(E) \;&:=\; \{\pi^E_E(s) \mid (\forall e \in E)\ \pi^E_{\{e\}}(s) = \boxed{e}\} \\ &=\; \langle unpad^{-1}\rangle\langle bc^{-1}\rangle \bigcap_{e\in E}\langle\pi^E_{\{e\}}\rangle\boxed{e} \end{aligned}$$

so that $Interval(\{e\}) = \boxed{e}$, and for $e \neq e'$, the set $Interval(\{e,e'\})$ consists of thirteen strings, one per interval relation in (Allen, 1983). We can organize these strings as follows (Fernando, 2012). For any finite set E, a string $s \in Interval(E)$ determines a triple $\langle E, \bigcirc_s, \prec_s\rangle$ with binary relations on E of *overlap* $e \bigcirc_s e'$ when $\pi^E_{\{e,e'\}}(s)$ is one of the nine strings in $Interval(\{e,e'\})$ that contain the pair $\boxed{e,e'}$

$$\begin{aligned} \bigcirc_s \;:=\; \{(e,e') \mid \pi^E_{\{e,e'\}}(s) \in\ & (\boxed{e} + \boxed{e'} + \epsilon) \\ & \boxed{e,e'}(\boxed{e} + \boxed{e'} + \epsilon)\} \end{aligned}$$

and *precedence* $e \prec_s e'$ when $\pi^E_{\{e,e'\}}(s)$ is either $\boxed{e}\boxed{e'}$ or $\boxed{e}\boxed{\ }\boxed{e'}$

$$\prec_s \;:=\; \{(e,e') \mid \pi^E_{\{e,e'\}}(s) \in \boxed{e}\boxed{e'} + \boxed{e}\boxed{\ }\boxed{e'}\}$$

leaving the two other strings in $Interval(\{e,e'\})$ for $e' \prec_s e$. The triple $\langle E, \bigcirc_s, \prec_s\rangle$ satisfies the axioms for an *event structure* in the sense of (Kamp and Reyle, 1993), and conversely, every such event structure over E can be obtained as $\langle E, \bigcirc_s, \prec_s\rangle$ for some $s \in Interval(E)$.

6 Discussion: simplifying where possible

Few would argue against representing information as simply as possible. Among strings, there is nothing simpler than the null string ϵ, and ϵ is the starting point for refinements given by the system of string functions π^A_B insofar as

$$\epsilon \;=\; \pi^A_B(\alpha_1\cdots\alpha_n) \quad \text{for all } \alpha_1\cdots\alpha_n \in (2^A)^* \text{ such that } B \cap \bigcup_{i=1}^{n}\alpha_i = \emptyset\,.$$

We have taken pains above to motivate the construction of $\pi^A_B := \rho^A_B; bc; unpad$ from

(i) ρ^A_B, linked in section 3 to MSO and regular languages via the Büchi-Elgot-Trakhtenbrot theorem

and from

(ii) bc and *unpad*, linked in section 5 to event structures for the Russell-Wiener construction of temporal moments from events (or temporal intervals).

A familiar example is provided by a calendar year, represented as the string

$$\text{year}_{mo} \;:=\; \boxed{\text{Jan}}\boxed{\text{Feb}}\cdots\boxed{\text{Dec}}$$

of length 12, one box per month from the set $mo := \{\text{Jan, Feb, Mar, }\ldots\text{, Dec}\}$. A finer-grained representation is given by the string

$$\text{year}_{mo,dy} \;:=\; \boxed{\text{Jan,d1}}\boxed{\text{Jan,d2}}\cdots\boxed{\text{Jan,d31}}\ \boxed{\text{Feb,d1}}\cdots\boxed{\text{Dec,d31}}$$

of length 365, one box per day, featuring unordered pairs from mo and $dy := \{\text{d1, d2, \ldots, d31}\}$. As the homomorphisms ρ^A_B see only what is in B,

$$\rho^{mo\cup dy}_{mo}(\text{year}_{mo,dy}) \;=\; \boxed{\text{Jan}}^{31}\boxed{\text{Feb}}^{28}\cdots\boxed{\text{Dec}}^{31}$$

which $\pi^{mo\cup dy}_{mo}$ then compresses to

$$bc(\boxed{\text{Jan}}^{31}\boxed{\text{Feb}}^{28}\cdots\boxed{\text{Dec}}^{31}) \;=\; \text{year}_{mo}$$

making $\pi^{mo\cup dy}_{mo}(\text{year}_{mo,dy}) = \text{year}_{mo}$. If ρ^A_B provides the key to establishing the regularity of MSO-formulas, block compression bc captures the essence of the slogan "no time without change" behind the Russell-Wiener conception of time. Whereas ρ^A_B limits what can be observed to what is in B, bc minimizes the time (space) over which to make these observations. Note that there are finite-state transducers that compute ρ^A_B and bc (over a finite alphabet). Thus, we may form the inverse image of a regular language under either ρ^A_B or bc without worrying if the result is still regular. (It is.)

Were we to leave *unpad* out and make do with $bc^A_B := \rho^A_B; bc$, we need only start our bc^A_B-based refinements from the string $\boxed{\;}$ of length one consisting of the empty set (rather than ϵ, as in the case of π^A_B) insofar as

$$\boxed{\;} \;=\; bc^A_B(\alpha_1\cdots\alpha_n) \qquad \text{for all } \alpha_1\cdots\alpha_n \in (2^A)^+ \text{ such that } B\cap\bigcup_{i=1}^{n}\alpha_i = \emptyset\,.$$

Indeed, $\boxed{\;}$ has the advantage over ϵ of qualifying as a model of MSO, which ϵ does not, under the usual convention that models of predicate logic have non-empty domains.

And even if one were to construct temporal spans from (say, closed intervals of) the real line $\mathbb{R}$ as in (Klein, 2009), the string $\boxed{\;}$ is a fine (enough) representation of $\mathbb{R}$, unbroken and virgin. The influential analysis of tense and aspect in (Reichenbach, 1947) positions the speech s and described event e relative to a reference time r. For instance, in the simple past (e.g. *it rained*), r coincides with e but precedes s

$$\text{simplePast}(e,r,s) \;:=\; \boxed{e,r\,|\,s} + \boxed{e,r\,|\,|\,s}$$

while in the present perfect (e.g. *it has rained*), r comes after e but coincides with s

$$\text{presentPerfect}(e,r,s) \;:=\; \boxed{e\,|\,s,r} + \boxed{e\,|\,|\,s,r}.$$

Factoring out the reference time, the simple past and present perfect become identical

$$\rho^{\{e,r,s\}}_{\{e,s\}}(\text{simplePast}(e,r,s)) \;=\; \boxed{e\,|\,s} + \boxed{e\,|\,|\,s} \;=\; \rho^{\{e,r,s\}}_{\{e,s\}}(\text{presentPerfect}(e,r,s)).$$

$\boxed{e\,|\,s} + \boxed{e\,|\,|\,s}$ is a simple example of "the expression of time in natural language" relating "a clause-internal temporal structure to a clause-external temporal structure" (Klein, 2009, page 75). The clause-internal structure $\boxed{e}$ and clause-external structure $\boxed{s}$ can be far more complex, subject to elaborations from lexical and grammatical aspect. Elaborations in interval temporal logic made in (Dowty, 1979) are formulated in terms of strings in (Fernando, 2013).

The finiteness and discreteness of strings arguably mirrors the bounded granularity of natural language statements (rife with talk of "the next moment"). Boundaries drawn to analyze, for example, telicity become absurd if they separate arbitrarily close pairs of real numbers (as they would, applied to the real line). It is customary to view a model M as an *index* that the Carnap-Montague *intension* of a formula φ maps to one of two truth values, indicating whether or not $M \models \varphi$. But the construal of a string as an underspecified representation suggests viewing it not only as an index but also as an *extension* (or denotation) of φ (Fernando, 2011). In this connection, a proposal from (Bach, 1986) is worth recalling — namely, that we associate an event type such as KISSING with a function EXT(KISSING) that maps histories to subparts that are temporal manifestations of KISSING (with input histories as indices, and output manifestations as extensions). Relativizing notions of indices and extensions to a bounded granularity, it is natural to assume at the outset not indices but extensions, which are then enlarged, as required, to more detailed and larger indices. The relation EXT(KISSING) then becomes a relation between strings (contained in $\sqsupseteq$) which a finite-state transducer might compute. For refinements of granularity, we start with an undifferentiated piece (viz. ϵ or $\boxed{\;}$) rather than a multitude of fully fleshed out possible histories. From ϵ or $\boxed{\;}$, the ingredients we require are a set of auxiliary symbols, and a suitable system R^A_B of regular relations, for finite sets A and $B \subseteq A$ of auxiliary symbols, projecting indices over the alphabet 2^A to indices over 2^B — e.g., bc^A_B or, as in (Fernando, 2013), π^A_B.

References

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the Association for Computing Machinery*, 26(11):832–843.

F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider. 2003. *The Description Logic Handbook: Theory, Implementation, Applications*. Cambridge University Press.

Franz Baader and Carsten Lutz. 2006. Description logic. In Patrick Blackburn, Johan van Benthem, and Frank Wolter, editors, *The Handbook of Modal Logic*, pages 757–820. Elsevier.

Emmon Bach. 1986. Natural language metaphysics. In R. Barcan Marcus, G.J.W. Dorn, and P. Weingartner, editors, *Logic, Methodology and Philosophy of Science VII*, pages 573 – 595. Elsevier.

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI,Stanford,

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

Andrzej Ehrenfeucht and Paul Zeiger. 1976. Complexity measures for regular expressions. *J. Comput. Syst. Sci.*, 12(2):134–146.

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

Tim Fernando. 2011. Regular relations for temporal propositions. *Natural Language Engineering*, 17(2):163184.

Tim Fernando. 2012. A finite-state temporal ontology and event-intervals. Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing, pages 80–89, Donostia/San Sebastian (ACL archive).

Tim Fernando. 2013. Segmenting temporal intervals for tense and aspect. 13th Mathematics of Language meeting. Sofia, Bulgaria.

Michael J. Fischer and Richard E. Ladner. 1979. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211.

Wouter Gelade and Frank Neven. 2008. Succinctness of the complement and negation of regular expressions. In *Symposium on Theoretical Aspects of Computer Science*, pages 325–336.

Markus Holzer and Martin Kutrib. 2010. The complexity of regular(-like) expressions. In *Developments in Language Theory*, pages 16–30. Springer.

Mans Hulden. 2009. Regular expressions and predicate logic in finite-state language processing. In *Finite-State Methods and Natural Language Processing*, pages 82–97. IOS Press.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic*. Kluwer.

Wolfgang Klein. 2009. How time is encoded. In *The Expression of Time*, pages 39–82. Mouton De Gruyter.

Hans Reichenbach. 1947. *Elements of Symbolic Logic*. MacMillan Company, NY.

Alfred Tarski, Andrzej Mostowski, and Raphael Robinson. 1953. *Undecidable Theories*. North-Holland.

Wolfgang Thomas. 1997. Languages, automata and logic. In *Handbook of Formal Languages: Beyond Words*, volume 3, pages 389–455. Springer-Verlag.

Anssi Yli-Jyrä and Kimmo Koskenniemi. 2004. Compiling Contextual Restrictions on Strings into Finite-State Automata. In *Proceedings of the Eindhoven FASTAR Days*.