On regular languages over power sets

Tim Fernando
Trinity College Dublin, Ireland

ABSTRACT

The power set of a finite set is used as the alphabet of a string interpreting a sentence of Monadic Second-Order Logic so that the string can be reduced (in straightforward ways) to the symbols occurring in the sentence. Simple extensions to regular expressions are described matching the succinctness of Monadic Second-Order Logic. A link to Goguen and Burstall's notion of an institution is forged, and applied to conceptions within natural language semantics of time based on change. Various reductions of strings are described, along which models can be miniaturized as strings.

Keywords:
regular language, power set, MSO, institution

1 INTRODUCTION

Working with more than one alphabet is established practice in finite-state language processing, attested by the popularity of auxiliary symbols (e.g., Kaplan and Kay 1994; Beesley and Karttunen 2003; Yli-Jyrä and Koskenniemi 2004; Hulden 2009). To avoid choosing an alphabet prematurely, implementations commonly treat the alphabet Σ as a dynamic entity that is left underspecified before the finite automaton is constructed in full.[1] Fixing Σ is not always necessary to determine the language denoted by an expression. This is the case with regular expressions; the expression $\emptyset$ denotes the empty set for any alphabet Σ, and the expression ab denotes the singleton set $\{ab\}$ for any alphabet $\Sigma \supseteq \{a, b\}$. Beyond regular expressions, however, there are expressions that denote different languages given different choices of

[1] I am indebted to an anonymous referee for raising this point.

the alphabet Σ. Consider ab's negation (or complement) $\overline{ab}$, which denotes a language

$$\Sigma^* - \{ab\} \;=\; \{s \in \Sigma^* \mid s \neq ab\}$$

that is regular iff Σ is a finite set. To delay fixing Σ to some finite set is to leave open just what the denotation $\Sigma^* - \{ab\}$ of $\overline{ab}$ is. Relative to an alphabet Σ, a symbol c, understood as a string of length one, belongs to that denotation if and only if $c \in \Sigma$. (Σ contains *any* symbol, including c, in the open alphabet system implemented in Beesley and Karttunen 2003.)

Apart from negations, there are many more extensions to regular expressions describing denotations that vary with the choice of alphabet. Consider the sentences of Monadic Second-Order Logic (MSO), which, under a model-theoretic interpretation against strings, capture the regular languages, by a fundamental theorem due independently to Büchi, Elgot and Trakhtenbrot (e.g., Theorem 3.2.11, page 145 in Grädel 2007; Theorem 7.21, page 124 in Libkin 2010). Leaving the precise details of MSO for Section 2 below, suffice it to say (for now) that occurrences of a string symbol a are encoded in a unary predicate symbol P_a for an MSO-sentence such as $\forall x P_a(x)$, saying a occurs at every string position (satisfied by the string aaa but not by the string ab unless $a = b$). We can check if a string over any finite alphabet Σ (hereafter, a Σ-string) satisfies an MSO-sentence φ, but the computation gets costlier as Σ is enlarged. Surely, however, only the symbols that appear in φ matter in satisfying φ or its negation? To investigate this question, let the *vocabulary of* φ be the set

$$voc(\varphi) \;:=\; \{a \mid P_a \text{ occurs in } \varphi\}$$

of subscripts of unary predicate symbols appearing in φ. (For example, $\forall x P_a(x)$'s vocabulary $voc(\forall x P_a(x))$ is $\{a\}$.) Now the question is: can we not reduce satisfaction of φ by a Σ-string to satisfaction of φ by a $voc(\varphi)$-string? A simple form such a reduction might take is a function $f : \Sigma^* \to voc(\varphi)^*$ mapping a Σ-string s to a $voc(\varphi)$-string $f(s)$ that satisfies φ if and only if s does

$$s \models \varphi \iff f(s) \models \varphi. \tag{1}$$

Unfortunately, already for φ equal to $\forall x P_a(x)$ and Σ to $\{a, b\}$, it is clear no such function f can exist; the lefthand side of (1) fails for $s = ab$, whereas the righthand side cannot: $a^n \models \forall x P_a(x)$ for all integers $n \geq 0$. Evidently, $voc(\varphi)^*$ is too small to provide the variation necessary for the reduction (1). Enter $(2^{voc(\varphi)})^*$, where the power set 2^A of a set A is the set of all subsets of A. For any MSO-sentence φ and string $s = \alpha_1 \cdots \alpha_n$ of sets α_i, we intersect s componentwise with $voc(\varphi)$ for the $2^{voc(\varphi)}$-string

$$\rho_{voc(\varphi)}(\alpha_1 \cdots \alpha_n) \;:=\; (\alpha_1 \cap voc(\varphi)) \cdots (\alpha_n \cap voc(\varphi)).$$

Then for any finite set Σ, we let MSO_Σ be the set of MSO-sentences with vocabulary contained in Σ

$$MSO_\Sigma \;:=\; \{\varphi \mid \varphi \text{ is an MSO-sentence and } voc(\varphi) \subseteq \Sigma\}$$

and interpret sentences $\varphi \in MSO_\Sigma$ relative to 2^Σ-strings s using a binary relation $\models_\Sigma$ (defined in Section 2) such that

$$s \models_\Sigma \varphi \iff \rho_{voc(\varphi)}(s) \models_{voc(\varphi)} \varphi. \tag{2}$$

The subscripts Σ and $voc(\varphi)$ on $\models$ in the lefthand and righthand sides of (2) track the reduction effected by $\rho_{voc(\varphi)}$ but could otherwise be dropped, had we not already used $\models$ for the satisfaction relation mentioned in (1). Fixing φ's denotation relative to Σ as the set

$$\mathscr{L}_\Sigma(\varphi) \;=\; \left\{ s \in (2^\Sigma)^* \mid s \models_\Sigma \varphi \right\}$$

of 2^Σ-strings that $\models_\Sigma$-satisfy φ, we may conclude from (2) that

(†) whatever finite set Σ we use to fix the denotation of φ, it all comes down to $voc(\varphi)$.

Our argument for (†) via (2) rests on modifying MSO-satisfaction $\models$ as it is usually presented over Σ-strings (e.g., Libkin 2010) to one $\models_\Sigma$ over 2^Σ-strings. Without appealing to (†), which might be made precise some other way, we motivate the step from Σ to 2^Σ in our presentation of MSO-models in Section 2, showing, among other things, how that step clarifies what predication and quantification amount to on strings (essentially, preimages and images under $\rho_{voc(\varphi)}$).

Beyond MSO, the reduction (2) is an instance of a general condition built into an abstract model-theoretic approach to specification and programming based on *institutions* (Goguen and Burstall 1992). We adopt this perspective to generalize (2) in Section 3 from $\rho_{voc(\varphi)}$

to functions on strings of sets, manipulating not only the vocabulary but also the length of strings (yielding, at the limit, infinite strings). At the center of this perspective are declarative methods for specifying sets of strings over different alphabets. We focus on methods, including but not limited to MSO, where the alphabets are power sets 2^{Σ} of finite sets Σ.

A multiplicity of such alphabets is useful in the semantics of tense and aspect to measure time at different bounded granularities Σ, tracking finite sets of unary predicates named in Σ. Consider, for instance, Reichenbach's well-known account based on a reference time R, an event time E and a speech time S (Reichenbach 1947). We can picture various temporal relations between an event and a speech as strings of boxes that may or may not contain E or S. For example, the string $\boxed{E}\boxed{S}$ portrays S after E (much like a film or comic strip), which we can verbalize using the simple past or the present perfect, illustrated by (a) and (b) respectively (where the event with time E is Ed's exhalation).

(a) Ed exhaled.
(b) Ed has exhaled.

To represent the difference between (a) and (b), we bring the reference time R into the picture, expanding $\Sigma = \{E,S\}$ to $\Sigma = \{R,E,S\}$ with

(‡) $\boxed{R,E}\boxed{S}$ for the simple past (a), and
$\boxed{E}\boxed{R,S}$ for the present perfect (b),

where a box is drawn instead of the usual curly braces $\{,\}$ for a set construed as a symbol in a string of sets. The difference brought out in (‡) carries significance for anaphora (e.g., Kamp and Reyle 1993, where R is split many ways) and event structure (including an event's consequent state, in Moens and Steedman 1988). Both strings in (‡) can be constructed from simpler strings representing a Reichenbachian analysis of

(i) tense as a relation between R and S, with $\Sigma = \{R,S\}$ and

$\boxed{R}\boxed{S}$ for the past (a), and $\boxed{R,S}$ for the present (b)

and

(ii) aspect as a relation between R and E, with $\Sigma = \{R,E\}$ and

$\boxed{R,E}$ for the simple (a), and $\boxed{E}\boxed{R}$ for the perfect (b).

Complicating the picture, there are finer analyses of E into aspectual classes going back to Aristotle, Ryle and Vendler (e.g., Dowty 1979) that call for an expansion of $\Sigma = \{R,E,S\}$ to refine the level of granularity (Fernando 2014). A wide ranging hypothesis that the semantics of tense and aspect is finite-state is defended in Fernando (2015), deploying regular languages over power sets, of the kind described below.

Applications to temporal semantics aside, the reader expecting a discussion of finite-state methods applied to phonology, morphology and/or syntax should be warned that such a discussion has been left for someone competent in such matters to take up elsewhere. The present paper claims neither to be the first nor the last word on regular languages over power sets. Its aim simply is to show how to get a handle on the dependence of certain declarative methods on the choice of a finite set Σ of symbols by stepping up to the power set 2^{Σ} of Σ and reducing a string through some function $\rho_{voc(\varphi)}$ or other. MSO provides an obvious point of departure (Section 2), leading to further declarative methods (Section 3).

2 MSO AND RELATED EXTENSIONS OF REGULAR EXPRESSIONS

It is convenient to fix an infinite set Z of symbols a that can appear in unary predicate symbols P_a, from which sentences of MSO are formed. An MSO-sentence φ can have within it only finitely many unary predicate symbols P_a, allowing us to break MSO up into fragments given by finite subsets Σ of Z (no single one of which encompasses all of MSO). In addition to the P_a's, we assume a binary relation symbol S (for successors), from which we can form, for example, the MSO-sentence

$$\forall x\big(P_a(x) \supset \exists y(S(x,y) \wedge P_b(y))\big)$$

saying that every a-occurrence is succeeded by a b-occurrence. Formal definitions are given in Subsection 2.1 of a satisfaction relation $\models_\Sigma$ between (finite) MSO_Σ-models and MSO_Σ-sentences, built from MSO_Σ-formulas with free variables analyzed by suitable expansions of Σ. These expansions are undone by functions ρ_Σ on strings that arguably provide the key to predication and quantification over strings. Indeed, the ρ_Σ's pave an easy route to the regularity of MSO, as we show in Subsection 2.2. The functions can be tweaked for useful extensions in Subsection 2.3 of regular expressions, and declarative methods in Section 3 that, like our presentation of MSO via $\models_\Sigma$, meet abstract requirements from Goguen and Burstall (1992).

In what follows, we write $Fin(A)$ for the set of finite subsets of a set A. Often but not always, A is Z.

2.1 *MSO-models, formulas and satisfaction*

We restrict our attention to finite models, defining for any integer $n \geq 0$, $[n]$ to be the set of integers from 1 to n,

$$[n] \;:=\; \{1, 2, \ldots, n\}$$

and S_n to be the successor (next) relation from i to $i+1$ for $i \in [n-1]$

$$S_n \;:=\; \{(1,2), (2,3), \ldots, (n-1, n)\}\,.$$

Given $\Sigma \in Fin(Z)$, let us agree that an *MSO_Σ-model* M is a tuple

$$\langle [n], S_n, \{[\![P_a]\!]\}_{a\in\Sigma}\rangle$$

for some integer $n \geq 0$,[2] such that for each $a \in \Sigma$, $[\![P_a]\!]$ is a subset of $[n]$ interpreting the unary relation symbol P_a. For $A \subseteq \Sigma$, the *A-reduct of M* is the MSO_A-model $\langle [n], S_n, \{[\![P_a]\!]\}_{a\in A}\rangle$, keeping only the interpretations $[\![P_a]\!]$ for $a \in A$.

There is a simple bijection *str* from MSO_Σ-models to 2^Σ-strings, picturing an MSO_Σ-model $M = \langle [n], S_n, \{[\![P_a]\!]\}_{a\in\Sigma}\rangle$ as the 2^Σ-string $str(M) = \alpha_1 \cdots \alpha_n$ with

$$\alpha_i \;:=\; \{a \in \Sigma \mid i \in [\![P_a]\!]\} \qquad\qquad (\text{for } i \in [n]),$$

which inverts to

$$[\![P_a]\!] \;=\; \{i \in [n] \mid a \in \alpha_i\} \qquad\qquad (\text{for } a \in \Sigma).$$

For example, if $\Sigma = \{a, b\}$ and M is $\left\langle [4], S_4, \{[\![P_c]\!]\}_{c\in\Sigma}\right\rangle$ with $[\![P_a]\!] = \{1, 2\}$ and $[\![P_b]\!] = \{1, 3\}$, then

$$str(M) \;=\; \boxed{a,b}\boxed{a}\boxed{b}\boxed{}$$

(with α_i boxed, as noted in the introduction, to mark them out as string symbols). Strings of boxes with exactly one $a \in \Sigma$ embed Σ^* into $(2^\Sigma)^*$; let $\iota : \Sigma^* \to (2^\Sigma)^*$ map $a_1 \cdots a_n \in \Sigma^n$ to

$$\iota(a_1 \cdots a_n) \;:=\; \boxed{a_1}\cdots\boxed{a_n}.$$

[2] We follow Libkin (2010) in allowing a model to have an empty domain/universe.

An advantage in working with $(2^\Sigma)^*$ rather than Σ^* is that we can intersect a 2^Σ-string $\alpha_1 \cdots \alpha_n$ componentwise with any subset A of Σ for the 2^A-string

$$\rho_A(\alpha_1 \cdots \alpha_n) \;:=\; (\alpha_1 \cap A) \cdots (\alpha_n \cap A)$$

(generalizing $\rho_{voc(\varphi)}$ in the introduction). The A-reduct of the MSO$_\Sigma$-model given by the string $\alpha_1 \cdots \alpha_n$ is represented by $\rho_A(\alpha_1 \cdots \alpha_n)$; i.e., for any MSO$_\Sigma$-model M and MSO$_A$-model M',

$$\rho_A(\mathit{str}(M)) = \mathit{str}(M') \iff M' \text{ is the } A\text{-reduct of } M.$$

The difference between an MSO$_\Sigma$-model M and the string $\mathit{str}(M)$ is so slight that we can confuse M harmlessly with $\mathit{str}(M)$ and refer to a 2^Σ-string as an MSO$_\Sigma$-model.

To form MSO-formulas with free variables, let us fix an infinite set *Var* disjoint from Z, $\mathit{Var} \cap Z = \emptyset$, treating each $x \in \mathit{Var}$ as a first-order variable. Given finite subsets Σ of Z and V of *Var*, we define a *MSO*$_{\Sigma,V}$*-model* to be a $2^{\Sigma \cup V}$-string in which each $x \in V$ occurs exactly once, and collect these in the set $\mathit{Mod}_V(\Sigma)$

$$\mathit{Mod}_V(\Sigma) \;:=\; \left\{ s \in (2^{\Sigma \cup V})^* \mid (\forall x \in V)\; \rho_{\{x\}}(s) \in \boxed{}^* \boxed{x} \boxed{}^* \right\}.$$

We define the set $\mathit{MSO}_{\Sigma,V}$ of MSO$_\Sigma$-formulas φ with free variables in V by induction, alongside sets $\mathscr{L}_{\Sigma,V}(\varphi)$ of strings in $\mathit{Mod}_V(\Sigma)$ that satisfy φ, determining a satisfaction relation

$$\models_{\Sigma,V} \;\subseteq\; \mathit{Mod}_V(\Sigma) \times \mathit{MSO}_{\Sigma,V}$$

between strings $s \in \mathit{Mod}_V(\Sigma)$ and formulas $\varphi \in \mathit{MSO}_{\Sigma,V}$ according to

$$s \models_{\Sigma,V} \varphi \iff s \in \mathscr{L}_{\Sigma,V}(\varphi).$$

The inductive definition consists of six clauses.

(a) If $\{x, y\} \subseteq V$, then $x = y$ and $S(x, y)$ are in $\mathit{MSO}_{\Sigma,V}$, with $x = y$ satisfied by strings in $\mathit{Mod}_V(\Sigma)$ where x and y occur in the same position

$$\mathscr{L}_{\Sigma,V}(x = y) \;:=\; \left\{ s \in \mathit{Mod}_V(\Sigma) \mid \rho_{\{x,y\}}(s) \in \boxed{}^* \boxed{x, y} \boxed{}^* \right\}$$

and $S(x, y)$ satisfied by strings in $\mathit{Mod}_V(\Sigma)$ where x occurs immediately before y

$$\mathscr{L}_{\Sigma,V}(S(x, y)) \;:=\; \left\{ s \in \mathit{Mod}_V(\Sigma) \mid \rho_{\{x,y\}}(s) \in \boxed{}^* \boxed{x} \boxed{y} \boxed{}^* \right\}.$$

(b) If $a \in \Sigma$ and $x \in V$, then $P_a(x)$ is in $\mathit{MSO}_{\Sigma,V}$ and is satisfied by strings in $\mathit{Mod}_V(\Sigma)$ where the occurrence of x coincides with one of a

$$\mathscr{L}_{\Sigma,V}(P_a(x)) := \left\{s \in \mathit{Mod}_V(\Sigma) \mid \rho_{\{a,x\}}(s) \in \left\{\boxed{}, \boxed{a}\right\}^* \boxed{a,x} \left\{\boxed{}, \boxed{a}\right\}^*\right\}.$$

(c) If $\varphi \in \mathit{MSO}_{\Sigma,V}$ then so is $\neg\varphi$ with $\neg\varphi$ satisfied by strings in $\mathit{Mod}_V(\Sigma)$ that do not satisfy φ

$$\mathscr{L}_{\Sigma,V}(\neg\varphi) := \mathit{Mod}_V(\Sigma) - \mathscr{L}_{\Sigma,V}(\varphi).$$

(d) If φ and ψ are in $\mathit{MSO}_{\Sigma,V}$ then so is $\varphi \wedge \psi$ with $\varphi \wedge \psi$ satisfied by strings in $\mathit{Mod}_V(\Sigma)$ that satisfy both φ and ψ

$$\mathscr{L}_{\Sigma,V}(\varphi \wedge \psi) := \mathscr{L}_{\Sigma,V}(\varphi) \cap \mathscr{L}_{\Sigma,V}(\psi).$$

For quantification, we must be careful that a variable can be reused, as in

$$P_b(x) \wedge \exists x P_a(x),$$

which is equivalent to $P_b(x) \wedge \exists y P_a(y)$ since $\exists x P_a(x)$ and $\exists y P_a(y)$ are.[3] To cater for reuse of $q \in \mathit{Var} \cup Z$, we define an equivalence relation $\sim_q$ between strings s and s' of sets that differ at most on q, putting

$$s' \sim_q s \iff \hat{\rho}_q(s') = \hat{\rho}_q(s),$$

where the function $\hat{\rho}_q$ removes q from a string $\alpha_1 \cdots \alpha_n$ of sets

$$\hat{\rho}_q(\alpha_1 \cdots \alpha_n) := (\alpha_1 - \{q\}) \cdots (\alpha_n - \{q\}).$$

We can now state the last two clauses of our inductive definition of $\mathit{MSO}_{\Sigma,V}$ and $\mathscr{L}_{\Sigma,V}(\varphi)$.

(e) If $\varphi \in \mathit{MSO}_{\Sigma,V\cup\{x\}}$ then $\exists x\varphi$ is in $\mathit{MSO}_{\Sigma,V}$ with $\exists x\varphi$ satisfied by strings in $\mathit{Mod}_V(\Sigma)$ that are $\sim_x$-equivalent to strings in $\mathit{Mod}_{V\cup\{x\}}(\Sigma)$ satisfying φ :

$$\mathscr{L}_{\Sigma,V}(\exists x\varphi) := \left\{s \in \mathit{Mod}_V(\Sigma) \mid (\exists s' \in \mathscr{L}_{\Sigma,V\cup\{x\}}(\varphi))\ s' \sim_x s\right\},$$

which simplifies in case x is not reused

$$\mathscr{L}_{\Sigma,V}(\exists x\varphi) = \left\{\rho_{\Sigma\cup V}(s) \mid s \in \mathscr{L}_{\Sigma,V\cup\{x\}}(\varphi)\right\} \qquad \text{if } x \notin V.$$

[3] We can always avoid reuse in finite formulas, working with finitely many variables.

(f) If $\varphi \in MSO_{\Sigma\cup\{a\},V}$ then $\exists P_a \varphi$ is in $MSO_{\Sigma,V}$ with $\exists P_a \varphi$ satisfied by strings in $Mod_V(\Sigma)$ that are $\sim_a$-equivalent to strings in $Mod_V(\Sigma \cup \{a\})$ satisfying φ :

$$\mathscr{L}_{\Sigma,V}(\exists P_a \varphi) \;:=\; \left\{s \in Mod_V(\Sigma) \mid (\exists s' \in \mathscr{L}_{\Sigma\cup\{a\},V}(\varphi))\; s' \sim_a s\right\},$$

which simplifies in case P_a is not reused

$$\mathscr{L}_{\Sigma,V}(\exists P_a \varphi) \;=\; \left\{\rho_{\Sigma\cup V}(s) \mid s \in \mathscr{L}_{\Sigma\cup\{a\},V}(\varphi)\right\} \qquad \text{if } a \notin \Sigma.$$

We adopt the usual abbreviations: $\varphi \vee \psi$ for $\neg(\neg\varphi \wedge \neg\psi)$, $\forall x \varphi$ for $\neg\exists x \neg\varphi$, etc. Also, we render second-order quantification $\exists P_a$ as $\exists X$, writing $\exists X \varphi$ for $\exists P_a \varphi^X_a$ where a does not occur in φ, and φ^X_a is φ with P_a replacing every occurrence of X. For example, we can express $x < y$ as $\exists X(X(y) \wedge \neg X(x) \wedge \text{closed}(X))$ where $\text{closed}(X)$ abbreviates $\forall x \forall y (X(x) \wedge S(x,y) \supset X(y))$, which we can picture as

$$\mathscr{L}_{\{a\},\emptyset}(\text{closed}(P_a)) \;=\; \boxed{}^{*}\boxed{a}^{*}$$

for the picture

$$\begin{aligned}
&\mathscr{L}_{\emptyset,\{x,y\}}(\exists P_a(P_a(y) \wedge \neg P_a(x) \wedge \text{closed}(P_a))) \\
&\qquad = \left\{\rho_{\{x,y\}}(s) \mid s \in \mathscr{L}_{\{a\},\{x,y\}}(P_a(y) \wedge \neg P_a(x) \wedge \text{closed}(P_a))\right\} \\
&\qquad = \left\{\rho_{\{x,y\}}(s) \mid s \in \boxed{}^{*}\boxed{x}^{*}\boxed{}^{*}\boxed{a}^{*}\boxed{a,y}\boxed{a}^{*}\right\} \\
&\qquad = \boxed{}^{*}\boxed{x}\boxed{}^{*}\boxed{y}\boxed{}^{*}
\end{aligned}$$

of $x < y$.

Next comes the pay-off in interpreting MSO-sentences over not just Z-strings but strings of sets. An easy proof by induction on $\varphi \in MSO_{\Sigma,V}$ establishes

Proposition 1 *Let $\Sigma \in Fin(Z)$ and $V \in Fin(Var)$. Then for all sets $A \subseteq \Sigma$ and $U \subseteq V$,*

$$MSO_{A,U} \;\subseteq\; MSO_{\Sigma,V}$$

and for all $\varphi \in MSO_{A,U}$,

$$\mathscr{L}_{\Sigma,V}(\varphi) \;=\; \left\{s \in Mod_V(\Sigma) \mid \rho_{A\cup U}(s) \in \mathscr{L}_{A,U}(\varphi)\right\}.$$

To pick out $MSO_{\Sigma,V}$-formulas with *no* free variables, we let $V = \emptyset$ for the set

$$MSO_{\Sigma} \;=\; MSO_{\Sigma,\emptyset}$$

of MSO_Σ-sentences, and write $\models_\Sigma$ for $\models_{\Sigma,\emptyset}$, and $\mathscr{L}_\Sigma(\varphi)$ for $\mathscr{L}_{\Sigma,\emptyset}(\varphi)$ (where $\varphi \in \mathit{MSO}_\Sigma$). An immediate corollary to Proposition 1 is that for all $\varphi \in \mathit{MSO}_\Sigma$ and $s \in \mathit{Mod}_\emptyset(\Sigma) = (2^\Sigma)^*$,

$$s \models_\Sigma \varphi \iff \rho_{voc(\varphi)}(s) \models_{voc(\varphi)} \varphi \tag{2}$$

where $voc(\varphi)$ is the smallest subset A of Z such that $\varphi \in \mathit{MSO}_A$

$$voc(\varphi) \;=\; \bigcap \{A \in \mathit{Fin}(Z) \mid \varphi \in \mathit{MSO}_A\}$$

(sharpening the description of $voc(\varphi)$ in the introduction).

2.2 *Regularity*

For any finite sets A and B, the restriction

$$\rho_A^B \;:=\; \rho_A \cap \left((2^B)^* \times (2^B)^*\right)$$

of ρ_A to $(2^B)^*$ is a regular relation – i.e. computed by a finite-state transducer (with one state, mapping $\alpha \subseteq B$ to $\alpha \cap A$). For the preimage (or inverse image) of a language L under a relation R, we borrow the notation

$$\langle R \rangle\, L \;:=\; \left\{ s \mid (\exists s' \in L)\; sRs' \right\}$$

from dynamic logic, instead of $R^{-1}L$ which becomes awkward for long R's. We can then rephrase the definition of $\mathit{Mod}_V(\Sigma)$ as

$$\mathit{Mod}_V(\Sigma) \;=\; \bigcap_{x \in V} \left\langle \rho_{\{x\}}^{\Sigma \cup V} \right\rangle \boxed{}^* \boxed{x} \boxed{}^* . \tag{3}$$

Similarly we have

$$\mathscr{L}_{\Sigma,V}(S(x,y)) \;=\; \mathit{Mod}_V(\Sigma) \cap \left\langle \rho_{\{x,y\}}^{\Sigma \cup V} \right\rangle \boxed{}^* \boxed{x}\boxed{y} \boxed{}^* \quad \text{for } x, y \in V$$

and writing θ_A^B for the inverse of ρ_A^B,

$$\begin{aligned} \mathscr{L}_{\Sigma,V}(\exists x \varphi) \;&=\; \mathit{Mod}_V(\Sigma) \cap \left\langle \rho_{\Sigma \cup V - \{x\}}^{\Sigma \cup V} \right\rangle \left\langle \theta_{\Sigma \cup V - \{x\}}^{\Sigma \cup V \cup \{x\}} \right\rangle \mathscr{L}_{\Sigma, V \cup \{x\}}(\varphi) \\ &=\; \mathit{Mod}_V(\Sigma) \cap \left\langle \theta_{\Sigma \cup V}^{\Sigma \cup V \cup \{x\}} \right\rangle \mathscr{L}_{\Sigma, V \cup \{x\}}(\varphi) \qquad \text{for } x \notin V. \end{aligned}$$

As regular languages are closed under intersection, complementation and preimages under regular relations (which are themselves closed under inverses), it follows that

Proposition 2 *For every $\Sigma \in Fin(Z)$, $V \in Fin(Var)$ and $\varphi \in MSO_{\Sigma,V}$, the set $\mathscr{L}_{\Sigma,V}(\varphi)$ of strings in $Mod_V(\Sigma)$ that satisfy φ is a regular language.*

The aforementioned Büchi–Elgot–Trakhtenbrot theorem (BET) sidesteps free variables, making do with $MSO_\Sigma = MSO_{\Sigma,\emptyset}$ and a fragment $\models^\Sigma \subseteq \Sigma^* \times MSO_\Sigma$ of $\models_\Sigma \subseteq (2^\Sigma)^* \times MSO_\Sigma$ given by Σ-strings s and $\varphi \in MSO_\Sigma$ such that

$$s \models^\Sigma \varphi \iff \iota(s) \models_\Sigma \varphi$$

(recalling from Subsection 2.1 that $\iota(a_1 \cdots a_n) = \boxed{a_1} \cdots \boxed{a_n}$ for $a_1 \cdots a_n \in \Sigma^n$). A language $L \subseteq \Sigma^*$ is then characterized by BET as regular iff for some sentence $\varphi \in MSO_\Sigma$,

$$L = \left\{ s \in \Sigma^* \mid s \models^\Sigma \varphi \right\}.$$

There is a sense in which the difference between s and $\iota(s)$ is purely cosmetic; a simple one-state finite-state transducer computes ι. But the MSO$_\Sigma$-sentences valid in $\models^\Sigma$ need not be valid in $\models_\Sigma$; take the MSO$_\Sigma$-sentence

$$spec(\Sigma) \;:=\; \forall x \bigvee_{a \in \Sigma} \Big(P_a(x) \wedge \bigwedge_{a' \in \Sigma - \{a\}} \neg P_{a'}(x) \Big)$$

specifying in every string position x, exactly one symbol a from Σ. BET effectively presupposes $spec(\Sigma)$ to extract from $\varphi \in MSO_\Sigma$ the regular language $\{s \in \Sigma^* \mid \iota(s) \models_\Sigma \varphi\}$ over Σ, rather than the full regular language $\mathscr{L}_\Sigma(\varphi)$ over 2^Σ from Proposition 2. To represent a regular language over 2^Σ, BET provides a sentence *not* in MSO_Σ but in MSO_{2^Σ}, which we can translate into MSO_Σ by replacing every subformula $P_\alpha(x)$ (for $\alpha \subseteq \Sigma$) with the conjunction

$$\bigwedge_{a \in \alpha} P_a(x) \;\wedge \bigwedge_{a' \in \Sigma - \alpha} \neg P_{a'}(x)$$

in $MSO_{\Sigma,\{x\}}$ interpretable by $\models_{\Sigma,V}$.[4] Insofar as computations are carried out on syntactic representations (e.g., MSO-formulas) rather than on semantic models (designed largely as theoretical aids to understanding), the explosion from Σ to 2^Σ is computationally worrying in the syntactic step from MSO_Σ to MSO_{2^Σ} rather than in the semantic enrichment of Σ^* to $(2^\Sigma)^*$.

[4] Conversely, we can translate MSO_Σ to MSO_{2^Σ} by replacing subformulas $P_a(x)$, for $a \in \Sigma$, with the disjunction $\bigvee\{P_\alpha(x) \mid \alpha \subseteq \Sigma$ and $a \in \alpha\}$ in $MSO_{2^\Sigma,\{x\}}$.

Underlying Proposition 2 is a recipe from $MSO_{\Sigma,V}$ to the regular expressions

$$\mathcal{L}_{\emptyset,\{x,y\}}(x = y) \;=\; \boxed{}^{*}\boxed{x,y}\boxed{}^{*}$$
$$\mathcal{L}_{\emptyset,\{x,y\}}(S(x,y)) \;=\; \boxed{}^{*}\boxed{x}\boxed{y}\boxed{}^{*}$$
$$\mathcal{L}_{\{a\},\{x\}}(P_a(x)) \;=\; \left\{\boxed{},\boxed{a}\right\}^{*}\boxed{a,x}\left\{\boxed{},\boxed{a}\right\}^{*}$$

closed under conjunction, complementation and preimages under ρ_A^B and θ_A^B. These extended regular expressions are as succinct as the formulas in $MSO_{\Sigma,V}$ they represent (up to a constant factor). That said, if we take the example of *spec*(Σ), we can simplify the recipe for $\mathcal{L}_\Sigma(\mathit{spec}(\Sigma))$ considerably to the image of Σ^* under ι

$$\mathcal{L}_\Sigma(\mathit{spec}(\Sigma)) \;=\; \left\{\boxed{a} \mid a \in \Sigma\right\}^{*}$$

linear in the size of Σ (as opposed to *spec*(Σ) with quadratically many occurrences of the variable x). The representability of regular languages by regular expressions in general (i.e., Kleene's theorem) raises the question: what useful finite-state tools does MSO add to the usual regular operations? Apart from intersection and complementation (the usual extensions to regular expressions), one tool that MSO_Σ introduces is the idea of a string as a model, the proper formulation of which blows Σ up to its power set 2^Σ (to represent all finite MSO_Σ-models, whether or not they satisfy *spec*(Σ)). Exploiting that blow up, we can define regular relations such as ρ_A^B under which preimages of regular languages are also regular. We modify the relations ρ_A^B in the next subsection, Subsection 2.3, examining the MSO representation of accepting runs of a finite automaton, which is demonstrably more succinct than any available with regular expressions.

2.3 *Some parts and sorts*

Using sets as symbols provides a ready approach to meronymy (i.e., parts); we drop the subscript A on ρ_A for the non-deterministic relation $\trianglerighteq$ of componentwise inclusion between strings of the same length

$$\alpha_1 \cdots \alpha_n \trianglerighteq \beta_1 \cdots \beta_m \iff n = m \text{ and } \alpha_i \supseteq \beta_i \text{ for } i \in [n]$$

called *subsumption* in Fernando (2004). For example, $s \trianglerighteq \rho_A(s)$ for all strings s of sets. A part of reduced length can be obtained by truncating a string s from the front for a suffix s'

$$s \textit{ suffix } s' \iff (\exists s'')\ s = s''s'$$

or from the back for a prefix s'

$$s \textit{ prefix } s' \iff (\exists s'')\ s = s's''.$$

We can then compose the relations $\trianglerighteq$, *suffix* and *prefix* for a notion $\sqsupseteq$ of *containment*

$$\begin{aligned} s \sqsupseteq s' &\iff (\exists s_1, s_2)\ s \trianglerighteq s_1 \text{ and } s_1 \textit{ suffix } s_2 \text{ and } s_2 \textit{ prefix } s' \\ &\iff (\exists u, v)\ s \trianglerighteq us'v \end{aligned}$$

between strings of possibly different lengths. For every atomic $\mathrm{MSO}_{\Sigma,V}$-formula φ, the satisfaction set $\mathscr{L}_{\Sigma,V}(\varphi)$ consists of the strings in $Mod_V(\Sigma)$ with characteristic $\sqsupseteq$-parts, given as follows.

Proposition 3 *For all disjoint finite sets Σ and V,*

$$\begin{aligned} \mathscr{L}_{\Sigma,V}(x = y) &= Mod_V(\Sigma) \cap \langle \sqsupseteq \rangle \boxed{x, y} && \textit{for } x, y \in V \\ \mathscr{L}_{\Sigma,V}(S(x, y)) &= Mod_V(\Sigma) \cap \langle \sqsupseteq \rangle \boxed{x}\boxed{y} && \textit{for } x, y \in V \\ \mathscr{L}_{\Sigma,V}(P_a(x)) &= Mod_V(\Sigma) \cap \langle \sqsupseteq \rangle \boxed{a, x} && \textit{for } a \in \Sigma, \quad x \in V. \end{aligned}$$

Under Proposition 3, each set $\mathscr{L}_{\Sigma,V}(\varphi)$ is the intersection of $Mod_V(\Sigma)$ with a language $\langle \sqsupseteq \rangle s_\varphi$, where s_φ is a string of length ≤ 2 that pictures φ. The obvious picture of $x < y$ is the set $\boxed{x}\boxed{}^*\boxed{y}$ of arbitrarily long strings

$$\mathscr{L}_{\Sigma,V}(x < y) = Mod_V(\Sigma) \cap \langle \sqsupseteq \rangle \boxed{x}\boxed{}^*\boxed{y} \qquad \text{for } x, y \in V$$

which is nonetheless easier to visualize (if not read) than the $MSO_{\emptyset,\{x,y\}}$-formula

$$\exists X\ \ (X(y) \wedge \neg X(x) \wedge (\forall u, v)(X(u) \wedge S(u, v) \supset X(v)))$$

expressing $x < y$. To compress the language $\boxed{x}\boxed{}^*\boxed{y}$ to the string $\boxed{x}\boxed{y}$, we can replace containment $\sqsupseteq$ by *weak containment*

$$\succeq \ := \ \{(\alpha_1 \cdots \alpha_n, x_1 \cdots x_n) \mid x_i = \epsilon \text{ or } x_i \subseteq \alpha_i \text{ for } i \in [n]\}$$

with deletions (x_i equal to the empty string ϵ) allowed anywhere, not just in the front or back of $\alpha_1 \cdots \alpha_n$ or inside any box α_i. (For example, $\boxed{x,a}\boxed{}^n\boxed{y}\boxed{} \succeq \boxed{x}\boxed{y}$ for all integers $n \geq 0$.) Proposition 3 holds with $\sqsupseteq$ and $S(x,y)$ replaced by $\succeq$ and $x < y$ respectively

$$\begin{array}{rcll} \mathscr{L}_{\Sigma,V}(x=y) & = & \mathit{Mod}_V(\Sigma) \cap \langle\succeq\rangle \boxed{x,y} & \text{for } x,y \in V \\ \mathscr{L}_{\Sigma,V}(x<y) & = & \mathit{Mod}_V(\Sigma) \cap \langle\succeq\rangle \boxed{x}\boxed{y} & \text{for } x,y \in V \\ \mathscr{L}_{\Sigma,V}(P_a(x)) & = & \mathit{Mod}_V(\Sigma) \cap \langle\succeq\rangle \boxed{a,x} & \text{for } a \in \Sigma,\ x \in V. \end{array}$$

Whether the part relation R is $\sqsupseteq$ or $\succeq$,[5] what matters for the regularity of $\mathscr{L}_{\Sigma,V}(\varphi)$ is that the restriction of R to $(2^{\Sigma\cup V})^*$

$$R \ \cap \ ((2^{\Sigma\cup V})^* \times (2^{\Sigma\cup V})^*)$$

is computable by a finite-state transducer (for all finite sets Σ and V). Within $\mathit{Mod}_V(\Sigma)$ are part relations $\rho_{\{x\}}$ (for $x \in V$) revealed by the equation

$$\mathit{Mod}_V(\Sigma) \ = \ \bigcap_{x\in V} \left\langle \rho_{\{x\}}^{\Sigma\cup V} \right\rangle \boxed{}^*\boxed{x}\boxed{}^*. \tag{3}$$

Moving from MSO to finite automata, let us rewrite pairs Σ, V as pairs A, Q of disjoint finite sets A and Q, and define an (A,Q)-*automaton* to be a triple $\mathscr{A} = (\to_{\mathscr{A}}, F_{\mathscr{A}}, q_{\mathscr{A}})$ consisting of

(i) a set $\to_{\mathscr{A}}$ of triples in $Q \times A \times Q$ specifying $\mathscr{A}$-transitions (where we write $q \xrightarrow{a}_{\mathscr{A}} q'$ instead of $(q,a,q') \in \to_{\mathscr{A}}$)

(ii) a set $F_{\mathscr{A}} \subseteq Q$ of $\mathscr{A}$-final states, and

(iii) an $\mathscr{A}$-initial state $q_{\mathscr{A}} \in Q$.

Given an (A,Q)-automaton $\mathscr{A}$, an $\mathscr{A}$*-accepting run* is a string

$$\boxed{a_1,q_1}\boxed{a_2,q_2}\cdots\boxed{a_n,q_n} \ \in \ (2^{A\cup Q})^*$$

such that $q_{\mathscr{A}} \xrightarrow{a_1}_{\mathscr{A}} q_1$ and $q_n \in F_{\mathscr{A}}$ and

$$q_{i-1} \xrightarrow{a_i}_{\mathscr{A}} q_i \ \text{ for } \ 1 < i \leq n$$

[5] For the present purposes, we can take a *part relation* to be any fragment R of $\succeq$ (i.e., whenever sRs', $s \succeq s'$). Thus, ρ_A, *suffix*, *prefix*, $\sqsupseteq$ and $\succeq$ are all part relations.

(where for $n = 0$, the empty string ϵ is an $\mathscr{A}$-accepting run iff $q_{\mathscr{A}} \in F_{\mathscr{A}}$). Let *AccRuns*$(\mathscr{A})$ be the set of $\mathscr{A}$-accepting runs. Clearly, for all $s \in A^*$,

$$\mathscr{A} \text{ accepts } s \iff (\exists s' \in \mathit{AccRuns}(\mathscr{A}))\ \iota(s) = \rho_A(s')$$

(recalling $\iota(a_1 \cdots a_n) = \boxed{a_1} \cdots \boxed{a_n}$). That is, $\mathscr{A}$ accepts the language

$$\mathscr{L}(\mathscr{A}) \ = \ \langle \iota_A \rangle \left\langle \theta_A^{A \cup Q} \right\rangle \mathit{AccRuns}(\mathscr{A})$$

(recalling θ_A^B is the inverse of ρ_A^B). As for the set *AccRuns*$(\mathscr{A})$ of $\mathscr{A}$-accepting runs, we start by collecting strings of pairs from A and Q in

$$\mathit{Pairs}(A,Q) \ := \ \bigcup_{n \geq 0} \left\{ \boxed{a_1, q_1} \cdots \boxed{a_n, q_n} \mid a_1 \cdots a_n \in A^n \text{ and } q_1 \cdots q_n \in Q^n \right\}.$$

We refine *Pairs*(A,Q) to *AccRuns*$(\mathscr{A})$, taking into account

(i) the set *Init*$[\mathscr{A}]$ of strings that start with a pair a, q such that $q_{\mathscr{A}} \overset{a}{\rightsquigarrow}_{\mathscr{A}} q$

$$\mathit{Init}[\mathscr{A}] \ := \langle \mathit{prefix} \rangle \left\{ \boxed{a, q} \mid q_{\mathscr{A}} \overset{a}{\rightsquigarrow}_{\mathscr{A}} q \right\}$$

(ii) the set *Final*$[\mathscr{A}]$ of strings ending with an $\mathscr{A}$-final state

$$\mathit{Final}[\mathscr{A}] \ := \ \langle \trianglerighteq \rangle \, \langle \mathit{suffix} \rangle \left\{ \boxed{q} \mid q \in F_{\mathscr{A}} \right\}$$

and

(iii) the set *Bad*$[\mathscr{A}]$ of strings containing $\boxed{q}\boxed{a, q'}$ for triples (q, a, q') *out*side the set $\rightsquigarrow_{\mathscr{A}}$ of $\mathscr{A}$-transitions

$$\mathit{Bad}[\mathscr{A}] \ := \ \langle \trianglerighteq \rangle \, \langle \mathit{suffix} \rangle \, \langle \mathit{prefix} \rangle \left\{ \boxed{q}\boxed{a, q'} \mid (q, a, q') \in Q \times A \times Q \text{ and not } q \overset{a}{\rightsquigarrow}_{\mathscr{A}} q' \right\}.$$

Note that $\langle R \rangle \, \langle R' \rangle \, L \ = \ \langle R; R' \rangle \, L$ for all relations R and R' and sets L, where $R; R'$ is the *relational composition of* R *and* R'

$$R; R' \ := \ \left\{ (s, s') \mid (\exists s'') \ sRs'' \text{ and } s''R's' \right\}$$

(and containment $\sqsupseteq$ is the relational composition of $\trianglerighteq$, *suffix* and *prefix*).

Proposition 4 *For all disjoint finite sets A and Q, and all (A,Q)-automata $\mathscr{A}$, the set AccRuns($\mathscr{A}$) of $\mathscr{A}$-accepting runs consists of all strings in Pairs(A,Q) that belong to Init[$\mathscr{A}$] and Final[$\mathscr{A}$] but not to Bad[$\mathscr{A}$]*

$$\mathit{AccRuns}(\mathscr{A}) \;=\; \mathit{Pairs}(A,Q) \cap \mathit{Init}[\mathscr{A}] \cap \mathit{Final}[\mathscr{A}] - \mathit{Bad}[\mathscr{A}].$$

Note that the language *Pairs*(A,Q) can be formed by defining for any finite sets C and D, the set

$$\mathit{Spec}_D(C) \;:=\; \mathscr{L}_{C\cup D}(\mathit{spec}(C)) \;=\; \langle \rho_C^{C\cup D} \rangle \left\{ \boxed{c} \mid c \in C \right\}^*$$

of $2^{C\cup D}$-strings with exactly one element of C in each box, making

$$\mathit{Pairs}(A,Q) \;=\; \mathit{Spec}_Q(A) \cap \mathit{Spec}_A(Q).$$

The language $\left\{ \boxed{c} \mid c \in C \right\}$ of ρ_C-parts of strings in $\mathit{Spec}_D(C)$ includes strings of any finite length, whereas all strings $\boxed{a,q}$, $\boxed{q}$ and $\boxed{q}\boxed{a,q'}$ pictured in $\mathit{Init}_{\mathscr{A}}$, $\mathit{Final}_{\mathscr{A}}$ and $\mathit{Bad}_{\mathscr{A}}$ have length ≤ 2. This is one sense in which the constraint *Pairs*(A,Q) is global (wide), while *Init*[$\mathscr{A}$] $\cap$ *Final*[$\mathscr{A}$]$-$*Bad*[$\mathscr{A}$] is local (narrow). A second sense is that *Pairs*(A,Q) captures accepting runs of all (A,Q)-automata, just as $\mathit{Mod}_V(\Sigma)$ in Proposition 3 captures all $\mathrm{MSO}_{\Sigma,V}$-models. That is, *Pairs*(A,Q) and $\mathit{Mod}_V(\Sigma)$ are general, sortal constraints that provide a context (or background) for more specific constraints to differentiate strings of the same sort; this differentiation is effected in Propositions 4 and 3 by attributes or parts that pick out substrings of length bounded by 2. Table 1 outlines the situation.

Table 1:

	sortal (taxonomic)	differential (meronymic)
Proposition 3	$\mathit{Mod}_V(\Sigma)$	$\langle \sqsupseteq \rangle\, s_\varphi$
Proposition 4	*Pairs*(A,Q)	$\mathit{Init}[\mathscr{A}] \cap \mathit{Final}[\mathscr{A}] - \mathit{Bad}[\mathscr{A}]$
	general	specific (to φ, $\mathscr{A}$)
length of part	unbounded (ρ_A)	bounded (≤ 2)

A further difference between the second and third columns of Table 1 is that whereas the sortal constraints $\mathit{Mod}_V(\Sigma)$ and *Pairs*(A,Q) employ deterministic part relations ρ_A, the differential constraints $\langle \sqsupseteq \rangle\, s_\varphi$ and $\mathit{Init}[\mathscr{A}] \cap \mathit{Final}[\mathscr{A}] - \mathit{Bad}[\mathscr{A}]$ employ non-deterministic relations $\sqsupseteq$, *prefix* and the relational composition $\trianglerighteq$; *suffix*. Although it is clear from Subsection 2.1 that the work done by $\sqsupseteq$, *prefix* and $\trianglerighteq$; *suffix* can be done by ρ_A, non-determinism nevertheless arises when introducing existential quantification through the inverse θ_A^B of ρ_A^B (used for the step from $\mathscr{A}$-accepting runs to the language $\mathscr{L}(\mathscr{A})$ accepted by $\mathscr{A}$). But while $\sqsupseteq$, *prefix* and $\trianglerighteq$; *suffix* search inside a string, θ_A^B searches outside. The search by θ_A^B is bounded only because the set B (that serves as its superscript) is finite (with elements of B not in A amounting to auxiliary symbols).

Non-determinism aside, the relations $\sqsupseteq$, *prefix* and $\trianglerighteq$; *suffix* differ from ρ_A and its inverse in relating strings of different lengths. Indeed, Table 1 arose above from the observation that parts with length ≤ 2 suffice for the constraints in the third column. That said, in the next section, we compress strings deterministically without setting any predetermined bounds (such as 2) on the resulting length, for sorts and parts alike.

3 COMPRESSION AND INSTITUTIONS

Having established through Proposition 1 the reduction

$$s \models_{\Sigma} \varphi \iff \rho_{voc(\varphi)}(s) \models_{voc(\varphi)} \varphi \tag{2}$$

(for all $\varphi \in MSO_{\Sigma}$ and $s \in (2^{\Sigma})^*$), we proceeded to part relations other than ρ_A in Table 1. The present section calls attention to string functions that can (unlike ρ_A) shorten a string, pointing the equivalence (2) and Table 1 in the direction of institutions (Goguen and Burstall 1992). As the length n of a string determines the domain $[n] = \{1, \ldots, n\}$ of the model encoded by the string, compression alters ontology over and above A-reducts produced by ρ_A.

3.1 *From compression to inverse limits*

We can strip off empty boxes at the front and back of a string s by defining

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

so that $unpad(s)$ neither begins nor ends with $\boxed{}$, making

$$\boxed{}^* \boxed{x}\,\boxed{}^* \;=\; \langle unpad \rangle \boxed{x}.$$

Using *unpad*-preimages, we can eliminate Kleene stars from the right side of

$$Mod_V(\Sigma) \ = \ \bigcap_{x \in V} \left\langle \rho^{\Sigma \cup V}_{\{x\}} \right\rangle \boxed{}^* \boxed{x} \boxed{}^* \tag{3}$$

and from the extended regular expressions from Proposition 3 for the sets $\mathscr{L}_{\Sigma,V}(\varphi)$ of strings satisfying formulas $\varphi \in MSO_{\Sigma,V}$. Regular expressions with complementation instead of Kleene star are known in the literature as *star-free regular expressions*, denoting, by a theorem of McNaughton and Papert, the first-order definable sets (Theorem 7.26, page 127, Libkin 2010). We can formulate a notion of Σ-*extended star-free expressions* matching the regular expressions over 2^Σ, but while it is easy enough to introduce the constructs $\langle \sqsupseteq \rangle$ and $\langle unpad \rangle$, we need subsets and supersets of Σ to relativize complementation and define the constructs $\left\langle \rho^B_A \right\rangle$ and $\left\langle \theta^B_A \right\rangle$, where θ^B_A is the inverse of ρ^B_A. On the positive side, this complication is potentially interesting as it suggests a hierarchy between the star-free regular languages and regular languages over 2^Σ. Be that as it may, our present concerns lie elsewhere.

Rather than separating the set *Var* of first-order variables from the set Z of subscripts a on unary predicates P_a, we can formulate the requirement on a symbol a that it occur exactly once in $MSO_{\{a\}}$

$$nom(a) \ := \ \exists x \forall y (P_a(y) \equiv x = y)$$

characteristic of *nominals* in the sense of Hybrid Logic (e.g., Braüner 2014, or “world variables” in Prior 1967, pages 187–197), with

$$\mathscr{L}_{\{a\}}(nom(a)) \ = \ \langle unpad \rangle \boxed{a}.$$

From $nom(a)$, it is a small step to the condition *interval*(a) that a occur in a string without gaps, which we can express in $\mathrm{MSO}_{\{a\}}$ as

$$interval(a) \ := \ \exists x \ P_a(x) \ \wedge \ \neg \exists y \ gap_a(y)$$

where $gap_a(y)$ says a does not occur at position y even though it occurs before and after y

$$gap_a(y) \ := \ \neg P_a(y) \wedge \exists u \exists v \ (u < y \wedge y < v \wedge P_a(u) \wedge \ P_a(v))$$

so that

$$\mathscr{L}_{\{a\}}(interval(a)) \ = \ \langle unpad \rangle \ \boxed{a}^+. \tag{4}$$

We can eliminate $\cdot^{+}$ from the right of (4) by defining a function *bc* that given a string s, compresses blocks α^n of $n > 1$ consecutive occurrences in s of the same symbol α to a single α, leaving s otherwise unchanged

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

so that $\boxed{a}^{+}$ is $\langle bc \rangle \boxed{a}$. In general, *bc* outputs only stutter-free strings, where a string $\alpha_1\alpha_2\cdots\alpha_n$ is *stutter-free* if $\alpha_i \neq \alpha_{i+1}$ for i from 1 to $n-1$. Construing boxes in a string as moments of time, we can view *bc* as implementing "McTaggart's dictum that 'there could be no time if nothing changed"' (Prior 1967, page 85). The restriction of *bc* to any finite alphabet is computable by a finite-state transducer, as are, for all $\Sigma \in Fin(Z)$ and $A \subseteq \Sigma$, the composition $\rho_A^\Sigma; bc$ for bc_A^Σ

$$bc_A^\Sigma(s) \;:=\; bc\big(\rho_A^\Sigma(s)\big) \qquad \text{for } s \in (2^\Sigma)^*$$

and the composition $bc_A^\Sigma; unpad$ for π_A^Σ

$$\pi_A^\Sigma(s) \;:=\; unpad\big(bc_A^\Sigma(s)\big) \;=\; bc\big(unpad(\rho_A^\Sigma(s))\big) \qquad \text{for } s \in (2^\Sigma)^*.$$

For $a \in \Sigma$, the (2^Σ)-strings in which a is an interval are those that $\pi_{\{a\}}^\Sigma$ maps to $\boxed{a}$

$$\mathscr{L}_\Sigma(interval(a)) \;=\; \left\langle \pi_{\{a\}}^\Sigma \right\rangle \boxed{a}.$$

The functions π_A^Σ compose nicely

$$\text{whenever } A \subseteq B \subseteq \Sigma, \quad \pi_A^\Sigma \;=\; \pi_B^\Sigma; \pi_A^B \tag{5}$$

from which it follows that

$$\begin{aligned} \mathscr{L}_\Sigma\big(\bigwedge_{a\in A} interval(a)\big) &= \bigcap_{a\in A} \mathscr{L}_\Sigma(interval(a)) \\ &= \bigcap_{a\in A} \left\langle \pi_{\{a\}}^\Sigma \right\rangle \boxed{a} \\ &= \left\langle \pi_A^\Sigma \right\rangle Interval(A) \end{aligned}$$

where *Interval*(A) is the π_A^A-image of $\bigcap_{a\in A} \left\langle \pi_{\{a\}}^A \right\rangle \boxed{a}$

$$Interval(A) \;:=\; \left\{ \pi_A^A(s) \mid s \in \bigcap_{a\in A} \left\langle \pi_{\{a\}}^A \right\rangle \boxed{a} \right\}.$$

Conflating a string s with the language $\{s\}$, observe that *Interval*$(\{a\}) = \boxed{a}$. For $a \neq a'$, the set *Interval*$(\{a, a'\})$ consists of thirteen strings, one per interval relation in Allen (1983), which can be partitioned

$$\mathit{Interval}(\{a,a'\}) \;=\; \mathscr{L}\left(a \bigcirc a'\right) \;\cup\; \mathscr{L}(a \prec a') \;\cup\; \mathscr{L}(a' \prec a)$$

between the nine-element set

$$\mathscr{L}\left(a \bigcirc a'\right) \;:=\; \left\{\boxed{a},\ \boxed{a'},\ \epsilon\right\}\boxed{a,a'}\left\{\boxed{a},\ \boxed{a'},\ \epsilon\right\}$$

describing overlap $\bigcirc$ between a and a' insofar as for all $s \in \mathit{Interval}(\Sigma)$ with $a, a' \in \Sigma$,

$$s \models_\Sigma \exists x\,(P_a(x) \wedge P_{a'}(x)) \iff \pi^{\Sigma}_{\{a,a'\}}(s) \in \mathscr{L}\left(a \bigcirc a'\right)$$

and the two-element sets

$$\mathscr{L}(a \prec a') \;:=\; \left\{\boxed{a}\boxed{a'},\ \boxed{a}\boxed{}\boxed{a'}\right\}$$
$$\mathscr{L}(a' \prec a) \;:=\; \left\{\boxed{a'}\boxed{a},\ \boxed{a'}\boxed{}\boxed{a}\right\}$$

describing complete precedence $\prec$ insofar as for all $s \in \mathit{Interval}(\Sigma)$ with $a, a' \in \Sigma$,

$$s \models_\Sigma \forall x \forall y\big((P_a(x) \wedge P_{a'}(y)) \supset x < y\big) \iff \pi^{\Sigma}_{\{a,a'\}}(s) \in \mathscr{L}(a \prec a')$$

and similarly for $a' \prec a$. Event structures are built around the relations $\bigcirc$ and $\prec$ in Kamp and Reyle (1993) (pages 667–674) to express the Russell-Wiener event-based conception of time, a particular elaboration of McTaggart's dictum mentioned above. The sets *Interval*(A) above provide representations of finite event structures (Fernando 2011).

Requiring that event structures be finite flies against the popularity of, for instance, the real line $\mathbb{R}$ in temporal semantics (e.g., Kamp and Reyle 1993, page 670). But we can approximate any infinite set Z by its set *Fin*(Z) of finite subsets, using the inverse system $(\mathit{Interval}(A))_{A \in Fin(Z)}$,

$$\pi_{A,B} : \mathit{Interval}(B) \to \mathit{Interval}(A), \quad s \mapsto \pi^{B}_{A}(s) \qquad \text{for } A \subseteq B \in \mathit{Fin}(Z)$$

for the *inverse limit*

$$\{\mathbf{a} : \mathit{Fin}(Z) \to \mathit{Fin}(Z)^* \mid \mathbf{a}(A) = \pi_{A,B}(\mathbf{a}(B)) \text{ whenever } A \subseteq B \in \mathit{Fin}(Z)\}$$

consisting of maps $\mathbf{a} : Fin(Z) \to Fin(Z)^*$ that respect the projections $\pi_{A,B}$. An element of that inverse limit, in case $\mathbb{R} \subseteq Z$, is the map $\mathbf{a}_{\mathbb{R}}$ such that for all $r_1 \cdots r_n \in \mathbb{R}^*$,

$$\mathbf{a}_{\mathbb{R}}(\{r_1, r_2, \ldots, r_n\}) = \boxed{r_1}\boxed{r_2}\boxed{}\cdots\boxed{r_n} \qquad \text{for } r_1 < r_2 < \cdots < r_n$$

copying $\mathbb{R}$. Notice that compressing strings via $\pi_{A,B}$ allows us to lengthen the strings in the inverse limit. If we remove the compression *bc* in $\pi_{A,B}$, we are left with the map ρ_A that leaves the ontology intact (insofar as the domain of an MSO-model is given by the string length), whilst restricting the vocabulary (for A-reducts).

3.2 *From inverse systems to institutions*

We have left out from the language $\mathit{Interval}(\{a\}) = \boxed{a}$ the string $\boxed{}\boxed{a}$ (among many others) that satisfies $\mathit{interval}(a)$, having built *unpad* into π^A_A. Notice that a is bounded to the left in $\boxed{}\boxed{a}$

$$\boxed{}\boxed{a} \models_{\{a\}} \exists x \exists y (S(x,y) \wedge P_a(y) \wedge \neg P_a(x))$$

but not in $\boxed{a}$. The functions π^B_A underlying $\mathit{Interval}(A)$ abstract away information about boundedness, which is fine if we assume intervals are bounded (as in Allen 1983). But what if we wish to study intervals that may or may not be left-bounded? Or, for that matter, strings where a may or may not be an interval? The line we pursue in this subsection harks back to Table 1 at the end of Section 2, encoding presuppositions in the second column (e.g., $Mod_V(\Sigma)$), and assertions in the third column (e.g., $\langle \sqsupseteq \rangle s_\varphi$). For instance, we presuppose a string s is stutter-free (i.e., $s = bc(s)$) and assert that a is an interval in s, to replace $\mathit{Interval}(A)$ by the intersection

$$\underbrace{\{bc(s) \mid s \in (2^A)^*\}}_{\text{presupposition}} \cap \underbrace{\bigcap \left\{ \left\langle \pi^A_{\{a\}} \right\rangle \boxed{a} \mid a \in A \right\}}_{\text{assertion}}$$

of which $\boxed{}\boxed{a}$ and $\boxed{a}$ are members, for $a \in A$. More generally, the idea is to refine the inverse system from the previous subsection to certain concrete instances of institutions (in the sense of Goguen and Burstall 1992) given by suitable functions on strings.

More precisely, let Z be a large set of symbols, and f be a function on *Fin*(Z)-strings (e.g., *bc*). For any finite subset A of Z, let $\mathsf{P}_f(A)$ be the image of $(2^A)^*$ under f

$$\mathsf{P}_f(A) \ := \ \{f(s) \mid s \in (2^A)^*\}$$

and let f_A be the composition $f_A = \rho_A; f$

$$f_A(s) \ := \ f(\rho_A(s)) \qquad \text{for } s \in \mathit{Fin}(Z)^*.$$

Thus, $\mathsf{P}_f(A)$ is the image of *Fin*$(Z)^*$ under f_A. More importantly, for every pair (B,A) of finite subsets of Z such that $A \subseteq B$, we define the function $\mathsf{P}_f(B,A) : \mathsf{P}_f(B) \to \mathsf{P}_f(A)$ sending $s \in \mathsf{P}_f(B)$ to $f_A(s) \in \mathsf{P}_f(A)$

$$\mathsf{P}_f(B,A)(s) \ := \ f_A(s) \qquad \text{for } s \in \mathsf{P}_f(B).$$

Now, to say P_f is an inverse system over *Fin*(Z) is to require that for all $A \in \mathit{Fin}(Z)$,

(c1) $\mathsf{P}_f(A,A)$ is the identity function on $\mathsf{P}_f(A)$; i.e.,

$$f_A(f(s)) \ = \ f(s) \qquad \text{for all } s \in (2^A)^*$$

and whenever $A \subseteq B \subseteq C \in \mathit{Fin}(Z)$,

(c2) $\mathsf{P}_f(C,A)$ is the composition $\mathsf{P}_f(C,B); \mathsf{P}_f(B,A)$; i.e.,

$$f_A(f(s)) \ = \ f_A(f_B(f(s))) \qquad \text{for all } s \in (2^C)^*.$$

Functions f validating conditions (c1) and (c2) include the identity function on *Fin*$(Z)^*$ (in which case f_A is ρ_A), *unpad* and *bc* (see Fernando 2014, where inverse systems P_f are referred to as presheaves). The condition (c2) reduces to the condition

$$\text{whenever } A \subseteq B \subseteq \Sigma, \quad \pi^{\Sigma}_{A} = \pi^{\Sigma}_{B}; \pi^{B}_{A} \tag{5}$$

from the previous subsection, for f equal to the composition *bc*; *unpad* (meeting also the requirement (c1)). To capture the entry $\mathit{Mod}_V(\Sigma)$ in the second column and row of Table 1 in terms of P_f, we must treat a first-order variable in V as a symbol $a \in Z$ (as in the previous subsection), and build into f both the uniqueness and existence conditions that *nom*(a) expresses, for $a \in V$. To ensure that no $a \in V$ occur more than once in a string s, we delete occurrences in s of a after its first, setting for all $\alpha_1 \cdots \alpha_n \in \mathit{Fin}(Z)^*$,

$$u_V(\alpha_1 \cdots \alpha_n) := \beta_1 \cdots \beta_n \qquad \text{where } \beta_i := \alpha_i - \Big(V \cap \bigcup_{j=1}^{i-1} \alpha_j\Big) \text{ for } i \in [n].$$

To ensure each $a \in V$ occurs at least once in the string, we put V at the very end

$$e_V(s\alpha) := s(\alpha \cup V)$$

with $e_V(\epsilon) := V$ for the empty string ϵ. Now, if f is the composition $e^V; u^V$ then

$$Mod_V(\Sigma) = \mathsf{P}_f(\Sigma \cup V)$$

and (c1) and (c2) hold.

The third column of Table 1 calls for further ingredients. Let us define a *Z-form* to be a function *sen* with domain $Fin(Z)$ mapping $A \in Fin(Z)$ to a set $sen(A)$ such that for all $B \in Fin(Z)$,

$$sen(A) \cap sen(B) \subseteq sen(A \cap B)$$

and

$$sen(A) \subseteq sen(B) \text{ whenever } A \subseteq B.$$

Given a Z-form *sen*, we can associate every $\varphi \in \bigcup\{sen(A) \mid A \in Fin(Z)\}$ with the finite subset

$$voc(\varphi) = \bigcap\{A \in Fin(Z) \mid \varphi \in sen(A)\}$$

of Z such that

$$\varphi \in sen(A) \iff voc(\varphi) \subseteq A$$

for all $A \in Fin(Z)$. Next, given a function f on $Fin(Z)^*$ and a Z-form *sen*, let us agree that a *(f, sen)-specification* $\mathscr{L}$ is a function with domain $Fin(Z)$ mapping $A \in Fin(Z)$ to a function $\mathscr{L}_A$ with domain $sen(A)$ mapping $\varphi \in sen(A)$ to a set $\mathscr{L}_A(\varphi)$ of strings in $\mathsf{P}_f(A)$. The intuition is that $\mathscr{L}_A(\varphi)$ consists of the strings in $\mathsf{P}_f(A)$ that A-satisfy φ

$$s \in \mathscr{L}_A(\varphi) \iff s\ A\text{-satisfies } \varphi \qquad (\text{for all } s \in \mathsf{P}_f(A)).$$

Putting the ingredients together, let us define a *(Z, f)-quadriplex* to be a 4-tuple $(Fin(Z), \mathsf{P}_f, sen, \mathscr{L})$ such that

(i) P_f is an inverse system over $Fin(Z)$

(ii) *sen* is a Z-form, and

(iii) $\mathscr{L}$ is a (f, sen)-specification.

Note that once Z and f are fixed, only the third and fourth components *sen* and $\mathscr{L}$ of a (Z, f)-quadriplex $(\mathit{Fin}(Z), \mathsf{P}_f, \mathit{sen}, \mathscr{L})$ may vary. To link up with institutions, as defined in Goguen and Burstall (1992), we view

(i) $\mathit{Fin}(Z)$ as a category with morphisms given by $\subseteq$

(ii) P_f as a contravariant functor from $\mathit{Fin}(Z)$ to the category **Set** of sets and functions, and

(iii) *sen* as a (covariant) functor from $\mathit{Fin}(\Phi)$ to **Set** such that whenever $A \subseteq B \in \mathit{Fin}(Z)$, $\mathit{sen}(A, B)$ is the inclusion $\mathit{sen}(A) \hookrightarrow \mathit{sen}(B)$.

The one remaining condition a (Z, f)-quadriplex must meet to be an institution is that for all $A \subseteq B \in \mathit{Fin}(Z)$ and $\varphi \in \mathit{sen}(A)$,

$$s \in \mathscr{L}_B(\varphi) \iff f_A(s) \in \mathscr{L}_A(\varphi) \qquad \text{(for all } s \in \mathsf{P}_f(B))$$

which we can put as the equation

$$\mathscr{L}_B(\varphi) = \mathsf{P}_f(B) \cap \langle f_A \rangle \ \mathscr{L}_A(\varphi).$$

In fact, the special case $A = \mathit{voc}(\varphi)$ suffices.

Proposition 5 *Given a set Z and function f on $\mathit{Fin}(Z)^*$, a (Z, f)-quadriplex $(\mathit{Fin}(Z), \mathsf{P}_f, \mathit{sen}, \mathscr{L})$ is an institution iff for all $\Sigma \in \mathit{Fin}(Z)$ and $\varphi \in \mathit{sen}(\Sigma)$,*

$$\mathscr{L}_\Sigma(\varphi) = \mathsf{P}_f(\Sigma) \cap \left\langle f_{\mathit{voc}(\varphi)} \right\rangle \ \mathscr{L}_{\mathit{voc}(\varphi)}(\varphi) \ . \tag{6}$$

If f is the identity on $\mathit{Fin}(Z)^*$, and $\mathit{sen}(\Sigma)$ is MSO_Σ, then (6) becomes the equivalence

$$s \models_\Sigma \varphi \iff \rho_{\mathit{voc}(\varphi)}(s) \models_{\mathit{voc}(\varphi)} \varphi \tag{2}$$

for all $\varphi \in \mathit{MSO}_\Sigma$ and $s \in (2^\Sigma)^*$. (6) also represents the division in Table 1 between column 2 ($\mathsf{P}_f(\Sigma)$) and column 3 ($\left\langle f_{\mathit{voc}(\varphi)} \right\rangle \mathscr{L}_{\mathit{voc}(\varphi)}(\varphi)$), whilst leaving open the possibility that f is not the identity function on $\mathit{Fin}(Z)^*$ nor is φ an MSO-formula.

Under (6), we may assume without loss of generality that *sen* and $\mathscr{L}$ have the following form. For every $\Sigma \in \mathit{Fin}(Z)$, there is a set $\mathrm{Expr}(\Sigma)$ of expressions e with denotations $[\![e]\!] \subseteq (2^\Sigma)^*$ such that $\mathit{sen}(\Sigma) = 2^\Sigma \times \mathrm{Expr}(\Sigma)$ consists of pairs (A, e) of subsets $A \subseteq \Sigma$ and $e \in \mathrm{Expr}(\Sigma)$ with $\mathit{voc}(A, e) = A$ and

$$\mathscr{L}_\Sigma(A, e) = \mathsf{P}_f(\Sigma) \cap \langle f_A \rangle \ [\![e]\!]. \tag{7}$$

An instructive example is provided by A equal to $\{a\}$, and e equal to the extended regular expression $\langle\sqsupseteq\rangle\, \boxed{a}\boxed{a}$ or equivalently, the MSO$_{\{a\}}$-sentence

$$\exists x \exists y \ (S(x,y) \wedge P_a(x) \wedge P_a(y)).$$

The righthand side of (7) can never hold with $f = bc$; there is *no* $s \in (2^\Sigma)^+$ such that $bc_{\{a\}}(s) \sqsupseteq \boxed{a}\boxed{a}$. A slight revision, however, makes the right hand side bc-satisfiable; introduce a symbol $b \neq a$ for A equal to $\{a,b\}$ and e equal to $\langle\sqsupseteq\rangle\, \boxed{a,b}\boxed{a}$ or the MSO$_{\{a,b\}}$-sentence

$$\exists x \exists y \ (S(x,y) \wedge P_a(x) \wedge P_a(y) \wedge P_b(x)).$$

In general, we can neutralize block compression bc on a string s by adding a fresh symbol to alternating boxes in s, which bc then leaves unchanged, since

$$bc(s) = s \iff s \text{ is stutter-free}$$

(recalling that $\alpha_1 \cdots \alpha_n$ is *stutter-free* if $\alpha_i \neq \alpha_{i+1}$ for $1 \leq i < n$). Similarly, we can add negations $\overline{a}$ of symbols a in A through a function $\boldsymbol{cl}_A$

$$\boldsymbol{cl}_A(\alpha_1 \cdots \alpha_n) \ := \ \beta_1 \cdots \beta_n \ \text{ where } \beta_i := \alpha_i \cup \{\overline{a} \mid a \in A - \alpha_i\} \text{ for } i \in [n]$$

to express bc^Σ_A in terms of π^Σ_B

$$bc^\Sigma_A \ = \ \boldsymbol{cl}_A; \pi^\Sigma_{c(A)}; \rho_A \quad \text{where} \ \ c(A) := A \cup \{\overline{a} \mid a \in A\}$$

treating $\overline{a} \in c(A) - A$ as an auxiliary symbol, and

$$bc^\Sigma_A; \boldsymbol{cl}_A \ = \ \boldsymbol{cl}_A; \pi^\Sigma_{c(A)}.$$

Returning to (7) with $f = bc$, we can say a is bounded to the left

$$\mathscr{L}_\Sigma(\{a\}, \exists x(\neg P_a(x) \wedge \forall y(P_a(y) \supset x < y))) \ = \ \left\langle bc^\Sigma_{\{a\}} \right\rangle \langle \mathit{prefix} \rangle \ \boxed{}$$

applying *prefix* after bc, and say a overlaps a'

$$\mathscr{L}_\Sigma(\{a,a'\}, \exists x(P_a(x) \wedge P_{a'}(x))) \ = \ \left\langle bc^\Sigma_{\{a,a'\}} \right\rangle \langle\sqsupseteq\rangle \ \boxed{a,a'}$$

applying containment $\sqsupseteq$ after bc. It is clear that *unpad* is just one of many relations that can come after bc^Σ_A (leading, in this case, to $\pi^\Sigma_A = bc^\Sigma_A; \mathit{unpad}$). The projection ρ^Σ_A in $bc^\Sigma_A = \rho^\Sigma_A; bc$ changes the granularity from Σ to A before bc reduces the ontology to suit A, and part relations (such as *prefix*, containment $\sqsupseteq$ or *unpad*) pick out a temporal span to frame a string (such as $\boxed{}$ or $\boxed{a, a'}$) picturing an assertion (e.g., left-boundeness, overlap). We are dividing here the choice of an expression e_φ denoting the language $\mathscr{L}_{voc(\varphi)}(\varphi)$ in Proposition 5 between a relation R and a string s for $e_\varphi = \langle R \rangle s$. Such a choice presupposes the finite approximability of the model of interest via the inverse limit of P_f (the discreteness of strings mirroring the bounded granularity of natural language statements, rife with talk of "the next moment"). Finite approximability is not only plausible but arguably implicit in accounts such as Reichenbach (1947) of tense and aspect.

4 CONCLUSION

There is no question that as declarative devices specifying sets of strings accepted by finite automata, regular expressions are more popular than MSO. What MSO offers, however, is a model-theoretic perspective on strings with computable notions of entailment (inclusions between regular languages being decidable), in addition to Boolean connectives that expose deficiencies in succinctness of regular expressions (e.g., Gelade and Neven 2012). Mapping a finite automaton $\mathscr{A}$ to a regular expression denoting the language $\mathscr{L}(\mathscr{A})$ accepted by $\mathscr{A}$ can have exponential cost (Ehrenfeucht and Zeiger 1976; Holzer and Kutrib 2010). A more concise representation of $\mathscr{L}(\mathscr{A})$ existentially quantifies away the internal states from the accepting runs of $\mathscr{A}$ (analyzed in Proposition 4 above). Not only can this be carried out in MSO (proving one half of the Büchi–Elgot–Trakhtenbrot theorem), but it is well-known that MSO-sentences can be far more succinct than finite automata (e.g., Libkin 2010, pages 124–125, and 135–136). To match the succinctness of MSO, regular expressions over alphabets 2^Σ (for finite sets Σ) are extended with preimages and images under homomorphisms ρ_A that output A-reducts, for $A \subseteq \Sigma$.

The step from Σ up to 2^Σ is justified by the various notions of part between strings of sets, given by ρ_A, subsumption $\trianglerighteq$, *prefix*, *suffix*, block compression *bc* and *unpad*, all computable (over 2^Σ) by finite-state transducers. Reducts between vocabularies are composed with compression within a fixed vocabulary to fit ontology against the vocabulary. An inverse limit construction (turning compression around to extension) takes us beyond the finite models of MSO to infinite timelines, approximated at granularity Σ by strings over the alphabet 2^{Σ}. Different finite sets Σ induce different notions $\models_{\Sigma}$ of satisfaction that form institutions, under certain minimal smoothness conditions (used to establish the Büchi–Elgot–Trakhtenbrot theorem in Section 2).

ACKNOWLEDGEMENTS

My thanks to Mark-Jan Nederhof for his editorship and four anonymous journal referees for their comments and help.

REFERENCES

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

Kenneth R. BEESLEY and Lauri KARTTUNEN (2003), *Finite State Morphology*, CSLI Publications, Stanford.

Torben BRAÜNER (2014), Hybrid Logic, The Stanford Encyclopedia of Philosophy, http://plato.stanford.edu/archives/spr2014/entries/logic-hybrid/.

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), Finite-state representations embodying temporal relations, in *Proceedings 9th International Workshop on Finite State Methods and Natural Language Processing*, pp. 12–20.

Tim FERNANDO (2014), Incremental semantic scales by strings, in *Proceedings EACL 2014 Workshop on Type Theory and Natural Language Semantics (TTNLS)*, pp. 63–71.

Tim FERNANDO (2015), The semantics of tense and aspect: A finite-state perspective, in S. LAPPIN and C. FOX, editors, *Handbook of Contemporary Semantic Theory*, pp. 203–236, Wiley-Blackwell, second edition.

Wouter GELADE and Frank NEVEN (2012), Succinctness of the complement and negation of regular expressions, *ACM Trans. Comput. Log.*, 13(1):4.1–4.19.

Joseph GOGUEN and Rod BURSTALL (1992), Institutions: Abstract model theory for specification and programming, *J. ACM*, 39(1):95–146.

Erich GRÄDEL (2007), Finite model theory and descriptive complexity, in *Finite Model Theory and Its Applications*, pp. 125–230, Springer.

Markus HOLZER and Martin KUTRIB (2010), The complexity of regular(-like) expressions, in *Developments in Language Theory*, pp. 16–30, Springer.

Mans HULDEN (2009), Regular expressions and predicate logic in finite-state language processing, in *Finite-State Methods and Natural Language Processing*, pp. 82–97, IOS Press.

Hans KAMP and Uwe REYLE (1993), *From Discourse to Logic*, Kluwer Academic Publishers, Dordrecht.

Ronald M. KAPLAN and Martin KAY (1994), Regular models of phonological rule systems, *Computational Linguistics*, 20(3):331–378.

Leonid LIBKIN (2010), *Elements of Finite Model Theory*, Springer.

Marc MOENS and Mark STEEDMAN (1988), Temporal ontology and temporal reference, *Computational Linguistics*, 14(2):15–28.

Arthur N. PRIOR (1967), *Past, Present and Future*, Clarendon Press, Oxford.

Hans REICHENBACH (1947), *Elements of Symbolic Logic*, London, Macmillan.

Anssi YLI-JYRÄ and Kimmo KOSKENNIEMI (2004), Compiling contextual restrictions on strings into finite-state automata, in *Proceedings of the Eindhoven FASTAR Days*.