Modular Temporal Logic

Augustin Baziramwabo* Université de Montréal

Pierre McKenzie* Université de Montréal

Denis Thérien† McGill University

Abstract

Thérien and Wilke characterized the Until hierarchy of linear temporal logic in terms of aperiodic monoids. Here, a temporal operator able to count modulo q is introduced. Temporal logic augmented with such operators is found decidable as it is shown to express precisely the solvable regular languages. Natural hierarchies are shown to arise when modular and conventional operators are interleaved. Modular operators are then cast as special cases of more general "group" temporal operators which, added to temporal logic, allow capturing any regular language L in much the same way that the syntactic monoid of L is constructed from groups and aperiodic monoids in the sense of Krohn-Rhodes.

Keywords: temporal logic, finite model theory, semigroup theory.

1 Introduction

In the last two decades, various systems of temporal logic have been used to specify and verify concurrent programs. The linear temporal logic approach defines the behavior of programs as their set of possible execution sequences. Since the program properties of interest typically have more to do with the flow of control than with the infinite number of possible values computed, it is adequate to assume a finite number of essentially different program states. Thus, a set of execution sequences can be regarded as a formal language over a finite alphabet.

Past temporal logic (PTL) refers to propositional logic supplemented with a unary "previous" operator $\ominus$, a unary "eventually in the past" operator $\Diamond$, and a binary "since" operator S (see Section 2). Kamp [6] proved thirty years ago that PTL interpreted over finite words precisely captures the star-free regular languages, which are also expressed by first-order logic (FO) as shown by Schützenberger [10] and McNaughton-Pappert [9].

The quantifier depth in FO and the "since" nesting depth in PTL both induce natural hierarchies of formulae. It is an amazing fact that, although FO and PTL are equally expressive in the end, they induce totally incomparable hierarchies: the quantifier depth in FO corresponds to the dot-depth hierarchy of Brzozowski-Knast [3, 15], and the "since" hierarchy in PTL was only recently characterized satisfactorily by Thérien and Wilke [13, 14], building on and streamlining work by Cohen, Perrin and Pin [4], and Etessami and Wilke [5].

The extent to which FO and PTL provide opposite viewpoints on the structure of star-free languages is in fact nicely explained by the algebraic characterizations of these hierarchies.

In this paper, we investigate an extension to PTL which we call *modular temporal logic* (PTL+MOD). We get PTL+MOD from PTL by introducing a new temporal operator able to count modulo an integer q. Our motivation for doing this is threefold. First, counting modulo q is a natural operation which could well be useful in temporal logic applications. Second, star-free languages have aperiodic syntactic semigroups, and cyclic counters are the simplest components which can serve to build more complicated semigroups. Third, a logic FO+MOD, defined from FO by introducing "modular quantifiers", was studied ten years ago [12] (see also [11]): in view of the opposite viewpoints afforded by FO and PTL, it is interesting to now compare the behaviors of counters in both logics.

*DIRO, Université de Montréal, Montréal (Québec), Canada, H3C 3J7 `{baziram, mckenzie}@iro.umontreal.ca`.

Work of the second author performed in part while on leave at Universität Tübingen. Supported by the (German) DFG, the (Canadian) NSERC and the (Québec) FCAR.

†School of Computer Science, McGill University, 3480 University Street, Montréal (Québec), Canada, H3A 2A7 `denis@cs.mcgill.ca`.

Our results are that a language is expressible in PTL+MOD iff it is regular and its syntactic monoid is solvable, i.e., contains no nonsolvable group. Hence, expressibility of a language in PTL+MOD is decidable. Moreover, a tight correspondence exists between alternations of abelian groups and aperiodic semigroups in Krohn-Rhodes's Decomposition Theorem [7] on the one hand, and the nesting of modular operators within levels of the usual past temporal logic on the other. Interestingly, although the operators used in FO and in PTL to "climb up" to the variety of aperiodic languages are very different (namely, the existential quantifiers in FO and the *Since* operator in PTL), the same MOD operator in both cases yields all solvable monoids.

We then sketch the relationship between yet a more expressive temporal logic, and the class of all regular languages. This more expressive temporal logic, which we call *group temporal logic*, is gotten by generalizing modular temporal operators further, to *group* temporal operators. In the presence of these operators, temporal logic can express any regular language L, and the formula used to express L is found to mimic the Krohn-Rhodes decomposition of the syntactic monoid of L.

Section 2 in this paper provides preliminaries and background on past temporal logic. Section 3 introduces modular temporal logic and proves our main characterization. Section 4 outlines the extension of these results to group temporal logic. A self-contained Appendix aims at providing intuition into a special case of our main theorem.

2 Preliminaries

A *past temporal logic formula* over a finite set A is built from the elements of A and the logical constant TRUE using the boolean connectives $\neg$ and $\vee$ and the usual temporal logic operators: $\ominus$ (previous), $\Diamond$ (eventually in 1the past), S (since). The former operators are unary while the latter is a binary operator.

Past temporal logic formulae are interpreted in strings over an alphabet Σ_A, the set of all subsets of A. We denote the length of a string $u \in \Sigma_A^+$ by $|u|$ and we number the positions in u from 0 to $|u|-1$. Given a string u and positions $i \leq j < |u|$, the substring $u(i) \cdots u(j)$ is denoted by $u[i,j]$.

Given a string $u \in \Sigma_A^+$ and a position $i < |u|$, we define what it means for a past temporal logic formula ϕ over A to be true in u at a time i, in symbols $(u,i) \models \phi$. We adopt the conventions of [14]:

$(u,i) \models \mathsf{TRUE}$,
$(u,i) \models a$ if $a \in u(i)$, where $a \in A$,
$(u,i) \models \neg\phi \Leftrightarrow$ not $(u,i) \models \phi$,
$(u,i) \models \phi \vee \psi$ if $(u,i) \models \phi$ or $(u,i) \models \psi$,
$(u,i) \models \ominus\phi$ if $i > 0$ and $(u,i-1) \models \phi$,
$(u,i) \models \Diamond\phi$ if there exists $j \leq i$ such that $(u,j) \models \phi$,
$(u,i) \models \phi \mathsf{S} \psi$ if there exists $0 \leq j \leq i$ such that $(u,j) \models \psi$ and, for all i', the condition $j < i' \leq i$ implies $(u,i') \models \phi$.

Given a past temporal logic formula ϕ, we write $u \models \phi$ and say that the string u is a model of ϕ if $(u,|u|-1) \models \phi$. We denote by $L(\phi)$ the set of all models of the formula ϕ, i.e., $L(\phi) = \{u \in \Sigma_A^+ \mid u \models \phi\}$ and we say that ϕ expresses the language $L(\phi)$. In a similar way, we say that a language $L \subseteq \Sigma_A^+$ is expressible in the past temporal logic if there exists a past temporal logic formula ϕ such that $L = L(\phi)$. Given a class of past temporal logic formulae Φ, we write $L(\Phi)$ for $\{L(\phi) \mid \phi \in \Phi\}$.

Kamp's deep result provided the first algebraic characterization of the past temporal logic:

Theorem 2.1 (Kamp [6]) *A language of Σ_A^+ is expressible in the past temporal logic if and only if its syntactic semigroup is aperiodic.*

Pseudovarieties of semigroups are classes of finite semigroups closed under quotients, finite direct products, and subsemigroups. If $\mathbf{V}$ is a pseudovariety, we write $L(\mathbf{V})$ the class of languages over some alphabet that are recognized by some elements from $\mathbf{V}$. We recall that a language $L \subseteq X^+$ is recognized by a semigroup M provided that there exist a morphism $\tau : X^+ \longrightarrow M$ and a subset $P \subseteq M$ such that $L = P\tau^{-1}$. For any language L, there exists a smallest –with respect to the property of "being morphic image of a subsemigroup of" and up to isomorphism– semigroup that recognizes it. This semigroup is called the *syntactic semigroup* of L.

Given semigroups S and T, the *wreath product* $S \circ T$ consists of all pairs (f,t) where $f \in S^T$ and $t \in T$. A multiplication in $S \circ T$ is defined by $(f_1,t_1)(f_2,t_2) = (g, t_1t_2)$, where for $t \in T$, $g(t) = f_1(t)f_2(t_1t)$. Given pseudovarieties $\mathbf{V}$ and $\mathbf{W}$ of semigroups, their wreath product, denoted by $\mathbf{V} \circ \mathbf{W}$, is the pseudovariety of semigroups generated by all semigroups of the form $S \circ T$ where $S \in \mathbf{V}$ and $T \in \mathbf{W}$. This product is associative. Alternatively, this operation on varieties can be defined in terms of the semidirect product: we prefer using wreath product because the languages recognized by $S \circ T$ are easier to describe than when the semidirect product is used. The pseudovarieties of abelian groups, of solvable groups, of groups, and of aperiodic semigroups, are denoted $\mathbf{G}_{com}$, $\mathbf{G}_{sol}$, $\mathbf{G}$, and $\mathbf{A}$ respectively.

A (synchronous) sequential function is a function $\sigma : X^+ \longrightarrow Y^+$ realized by a (synchronous) finite transducer. For the level of exposition of this abstract, we will only require the property that a synchronous sequential function is length-preserving.

3 Modular Temporal logic

By *modular temporal logic* (PTL+MOD), we will mean the past temporal logic of the previous section augmented with new unary temporal operators $\mathsf{Mod}_{j,q}$ for integers $0 \leq j < q$. The new modular operators have the following natural semantics: given a PTL+MOD formula ϕ,

$$(u,i) \models \mathsf{Mod}_{j,q}\phi$$

if, modulo q, there are j positions i', $0 \leq i' \leq i$, such that $(u,i') \models \phi$.

3.1 Substitution

Thérien and Wilke observed that replacing each occurrence of the atomic subformula a within a temporal logic formula ϕ by a formula ψ sometimes corresponds, from the algebraic point of view, to forming a wreath product. Developing this "semidirect product/substitution principle" was indeed the key to their results.

Much of our work will amount to verifying that this principle stills holds in our more general PTL+MOD setting. The algebraic tools underlying the principle can appear intimidating even to temporal logic experts, which we took as justification for the present paper. We will not, though, reprove the principle from scratch. Instead, we will assume that the reader has access to [14], and we will complement the proofs found there to account for the new modular operators.

If ϕ is a PTL+MOD formula over an alphabet A and $(\psi_a)_{a\in A}$ is a family of PTL+MOD formulae over B, then $\phi[a \mapsto \psi_a]$ stands for the PTL+MOD formula over B obtained from ϕ by replacing every occurrence of the subformula a in ϕ by the corresponding formula ψ_a. Formally, we define substitution as in [14]:

$$\begin{aligned}
\mathsf{TRUE}[a \mapsto \psi_a] &= \mathsf{TRUE}, \\
a'[a \mapsto \psi_a] &= \psi_{a'}, \forall a' \in A, \\
\neg\phi[a \mapsto \psi_a] &= \neg(\phi[a \mapsto \psi_a]), \\
\phi_1 \vee \phi_2[a \mapsto \psi_a] &= (\phi_1[a \mapsto \psi_a]) \vee (\phi_2[a \mapsto \psi_a]), \\
\ominus\phi[a \mapsto \psi_a] &= \ominus(\phi[a \mapsto \psi_a]), \\
\Diamond\phi[a \mapsto \psi_a] &= \Diamond(\phi[a \mapsto \psi_a]), \\
\phi_1 \mathbin{\mathsf{S}} \phi_2[a \mapsto \psi_a] &= (\phi_1[a \mapsto \psi_a]) \mathbin{\mathsf{S}} (\phi_2[a \mapsto \psi_a]), \\
(\mathsf{Mod}_{j,q}\phi)[a \mapsto \psi_a] &= \mathsf{Mod}_{j,q}(\phi[a \mapsto \psi_a]), \\
& \quad \forall q > j \geq 0.
\end{aligned}$$

Given a class Φ of PTL+MOD formulae over an alphabet A, and a class Ψ of PTL+MOD formulae, we write $\Phi \star \Psi$ for the class of PTL+MOD formulae that are boolean combinations of formulae from Ψ and of formulae that are boolean combinations of propositional variables and formulae $\phi[a \mapsto \psi_a]$, with $\phi \in \Phi$ and $\psi_a \in \Psi$, for every $a \in A$.

Obviously, modular operators preserve associativity of substitution [14]:

Proposition 3.1 (Associativity of substitution) *Let ϕ be a PTL+MOD formula over an alphabet A, $(\psi_a)_{a\in A}$ a family of PTL+MOD formulae over an alphabet B, and $(\xi_b)_{b\in B}$ a family of PTL+MOD formulae over an alphabet C. Then,*

$$\phi[a \mapsto \psi_a[b \mapsto \xi_b]] \quad = \quad \phi[a \mapsto \psi_a][b \mapsto \xi_b]. \quad (1)$$

We now wish to argue that the "substitution lemma for past temporal logic", found in [14], also holds for PTL+MOD. As in [14], for a family $(\psi_a)_{a\in A}$ of PTL+MOD formulae over B and a string $u \in \Sigma_B^+$, define a string $u[a \mapsto \psi_a]$ over Σ_A as follows: $u[a \mapsto \psi_a] = v_0 \cdots v_{|u|-1}$ where $v_i = \{a \in A \mid (u,i) \models \psi_a\}$. We then get:

Lemma 3.2 (substitution for PTL+MOD) *Let ϕ be a PTL+MOD formula over A and $(\psi_a)_{a\in A}$ a family of PTL+MOD formulae over B. For every string $u \in \Sigma_B^+$ and every position $i < |u|$, the following holds:*

$$(u,i) \models \phi[a \mapsto \psi_a] \quad \Leftrightarrow \quad (u[a \mapsto \psi_a], i) \models \phi. \quad (2)$$

Proof. The proof is by induction on the structure of ϕ. Here we only illustrate the inductive step for the modular operators. Let $\phi = \mathsf{Mod}_{j,q}\phi'$. Then $(u,i) \models \phi[a \mapsto \psi_a]$ holds

$$\begin{aligned}
\text{iff} \quad & |\{\ 0 \leq i' \leq i \ : \ (u,i') \models \phi'[a \mapsto \psi_a]\ \}| \\
& \equiv j \bmod q \\
\text{iff} \quad & |\{\ 0 \leq i' \leq i \ : \ (u[a \mapsto \psi_a], i') \models \phi'\ \}| \\
& \equiv j \bmod q \\
\text{iff} \quad & (u[a \mapsto \psi_a], i) \models \mathsf{Mod}_{j,q}\phi'
\end{aligned}$$

where the second assertion follows by the induction hypothesis. ∎

As a consequence of Lemma 3.2 we get: if ϕ and ϕ' are PTL+MOD formulae over A, and $(\psi_a)_{a\in A}$ and

$(\psi'_a)_{a\in A}$ are families of PTL+MOD formulae over B such that $L(\phi) = L(\phi')$ and $L(\psi_a) = L(\psi'_a)$ for every $a \in A$, then $L(\phi[a \mapsto \psi_a]) = L(\phi'[a \mapsto \psi'_a])$. There is a close relationship between substitution and inverse images of synchronous sequential functions [14]:

Lemma 3.3 *Let $\sigma : \Sigma_B^+ \longrightarrow \Sigma_A^+$ be a synchronous sequential function and let $(\psi_a)_{a\in A}$ be a family of PTL+MOD formulae over B such that, for every $a \in A$, $(L(a))\sigma^{-1} = L(\psi_a)$. Then, for every PTL+MOD formula ϕ over A,*

$$(L(\phi))\sigma^{-1} \quad = \quad L(\phi[a \mapsto \psi_a]). \tag{3}$$

Proof. The proof is by induction on the structure of ϕ. Again we only consider the modular operator, the other cases being dealt with in [14]. Let $q > j \geq 0$ and $\phi = \mathsf{Mod}_{j,q}\phi'$. Then, $u \in L(\phi[a \mapsto \psi_a])$ holds

$$\begin{array}{ll}
\text{iff} & u \in L(\mathsf{Mod}_{j,q}(\phi'[a \mapsto \psi_a])) \ \ \text{(substitution)} \\
\text{iff} & |\{\ 0 \leq i < |u| \ \ : \ \ (u,i) \models \phi'[a \mapsto \psi_a]\ \}| \\
& \equiv\ j \bmod q \ \ \text{(semantics)} \\
\text{iff} & |\{\ 0 \leq i < |u| \ \ : \ \ u[0,i] \in (L(\phi'))\sigma^{-1}\ \}| \\
& \equiv\ j \bmod q \ \ \text{(induction)} \\
\text{iff} & |\{\ 0 \leq i < |u| \ \ : \ \ (u\sigma)[0,i] \in L(\phi')\ \}| \\
& \equiv\ j \bmod q \ \ (\sigma \text{ synchronous}) \\
\text{iff} & u\sigma \models \mathsf{Mod}_{j,q}\phi' \ \ \text{(semantics)} \\
\text{iff} & u \in (L(\phi))\sigma^{-1} \ \ \text{(definition of } \phi).
\end{array}$$

■

If the substituents are very simple, the substitution operation can be described using morphisms. We do not reproduce the proof [14] that Lemma 3.3 implies Lemma 3.4 and Lemma 3.5 below:

Lemma 3.4 *Let ϕ be a PTL+MOD formula over A and $(\psi_a)_{a\in A}$ an A-indexed family of PTL+MOD formulae over B. Assume that each formula ψ_a is a disjunction of atomic formulae from B. Then, $L(\phi[a \mapsto \psi_a])$ is the preimage of $L(\phi)$ under some morphism $\Sigma_B^+ \longrightarrow \Sigma_A^+$.*

Lemma 3.5 *Let ϕ be a PTL+MOD formula over A and $(\psi_a)_{a\in A}$ an A-indexed family of PTL+MOD formulae over B. Furthermore, assume that each formula ψ_a is a boolean combination of atomic variables from B. Then, $L(\phi[a \mapsto \psi_a])$ is the preimage of $L(\phi)$ under a morphism $\Sigma_B^+ \longrightarrow \Sigma_A^+$.*

3.2 Semidirect product/substitution principle for PTL+MOD

Define the following classes of PTL+MOD formulae:

- Pv ("Previous"): formulae with no $\Diamond$, S, or $\mathsf{Mod}_{j,q}$ operators,
- PTL ("Past temporal logic"): formulae with no $\mathsf{Mod}_{j,q}$ operators.

Lemmas 3.2, 3.3, 3.4, and 3.5 imply Theorem 3.6 exactly as argued in [14]:

Theorem 3.6 (substitution principle)
Let $\boldsymbol{V}$, $\boldsymbol{W}$ be pseudovarieties of finite semigroups and Φ, Ψ classes of PTL+MOD formulae such that $L(\boldsymbol{V}) = L(\Phi)$ and $L(\boldsymbol{W}) = L(\Psi)$. If $\Phi \star \mathsf{Pv} \star \Psi \subseteq \Phi \star \Psi$, then

$$L(\boldsymbol{V} \circ \boldsymbol{W}) = L(\Phi \star \Psi). \tag{4}$$

We will make use of Theorem 3.6 in the following form:

Corollary 3.7 *Let $\boldsymbol{V}$ and $\boldsymbol{W}$ be pseudovarieties of finite semigroups and let Φ and Ψ be classes of PTL+MOD formulae such that $L(\boldsymbol{V}) = L(\Phi)$ and $L(\boldsymbol{W}) = L(\Psi)$. If, for some class Ξ of formulae, $\Phi = \Xi \star \mathsf{PTL}$ or $\Psi = \mathsf{PTL} \star \Xi$, then $L(\boldsymbol{V} \circ \boldsymbol{W}) = L(\Phi \star \Psi)$.*

Proof. Clearly, $\mathsf{PTL} \star \mathsf{Pv} = \mathsf{PTL} = \mathsf{Pv} \star \mathsf{PTL}$. Hence in the first case, i.e. if $\Phi = \Xi \star \mathsf{PTL}$, then $\Phi \star \mathsf{Pv} \star \Psi = \Xi \star \mathsf{PTL} \star \mathsf{Pv} \star \Psi = \Xi \star \mathsf{PTL} \star \Psi = \Phi \star \Psi$, so that Theorem 3.6 applies. The second case is analogous. ■

3.3 Algebraic characterization of PTL+MOD

Define the following classes of PTL+MOD formulae:

- Bool: the boolean closure of the atomic formulae a, $a \in A$.
- Mod: formulae with no operators other than $\mathsf{Mod}_{j,q}$ operators applied to Bool formulae.

Before stating our main result, we require an observation:

Lemma 3.8 *Let $q > j \geq 0$. Then*

1. *For any Bool formula ϕ, $L(\mathsf{Mod}_{j,q}\phi)$ is recognizable by Z_q.*
2. *If L is recognized by Z_q then L is a boolean combination of languages of the form $L(\mathsf{Mod}_{j,q}\phi)$ where ϕ is a Bool formula.*

Proof. To prove (1), consider a Bool formula ϕ and $q > j \geq 0$. A clear property of ϕ is that, for any $u \in \Sigma_A^+$ and any $0 \leq i < |u|$, whether (u,i) is a model of ϕ depends only on $u[i,i]$. Hence it suffices to map each $s \in \Sigma_A$ to 1 or 0 appropriately in order to ensure that each (u,i), $0 \leq i < |u|$, is counted correctly by

Z_q. Formally, define the morphism $\tau : \Sigma_A^+ \longrightarrow Z_q$ by setting, for each $s \in \Sigma_A$,

$$s\tau = \begin{cases} 1 & \text{if } (s,0) \models \phi \\ 0 & \text{otherwise.} \end{cases}$$

Then $j\tau^{-1} = \{u \in \Sigma_A^+ : |\{0 \le i < |u| : (u,i) \models \phi\}| \equiv j \bmod q\} = L(\mathsf{Mod}_{j,q}\phi)$. Therefore, $L(\mathsf{Mod}_{j,q}\phi)$ is recognized by Z_q.

To prove (2), let $\tau : \Sigma_A^+ \to Z_q$ recognize $L = L\tau\tau^{-1}$. It suffices to show that, for an arbitrary $j \in Z_q$, $j\tau^{-1}$ is a boolean combination of languages of the form $L(\mathsf{Mod}_{j,q}\phi)$ for some Bool formula ϕ. For each $s \in \Sigma_A$, define the formula

$$\mathrm{iam}(s) = (\bigwedge_{a \in s} a) \wedge (\bigwedge_{a \notin s} \neg a). \tag{5}$$

Let $s_1, s_2, \ldots, s_{|\Sigma_A|}$ be the elements of Σ_A. For every $|\Sigma_A|$-tuple $\vec{i} = \langle j_1, \cdots, j_{|\Sigma_A|}\rangle$ of integers each of which ranging over all Z_q, let

$$E(s_1, \cdots, s_{|\Sigma_A|}; \vec{i}) = (s_1\tau)j_1 + (s_2\tau)j_2 + \cdots + (s_{|\Sigma_A|}\tau)j_{|\Sigma_A|}. \tag{6}$$

Then $j\tau^{-1}$ is expressed by

$$\bigvee_{E(s_1, s_2, \cdots, s_{|\Sigma_A|}; \vec{i}) \equiv j \bmod q} \left(\bigwedge_{k=1}^{|\Sigma_A|} \mathsf{Mod}_{j_k,q}\mathrm{iam}(s_k) \right)$$

■

Our main result is the following theorem:

Theorem 3.9 *A language $L \subseteq \Sigma_A^+$ is expressible in PTL+MOD if and only if it is regular and its syntactic semigroup is solvable.*

Theorem 3.9 implies that PTL+MOD is decidable: to decide whether a (regular) language L is expressible in PTL+MOD, construct its syntactic semigroup and test it for solvability. Theorem 3.9 follows from the much sharper characterization stated as Lemma 3.10 below. Indeed, a language is regular and solvable iff its (finite) syntactic semigroup is in some pseudovariety $\mathbf{A} \circ \mathbf{G}_{com} \circ \mathbf{A} \circ \cdots \circ \mathbf{G}_{com} \circ \mathbf{A}$. Hence Lemma 3.10 directly implies Theorem 3.9 and yields much more information.

Lemma 3.10
1. $L((\boldsymbol{G}_{com} \circ \boldsymbol{A} \circ)^i\ \boldsymbol{G}_{com})$ $= L((\mathsf{Mod} \star \mathsf{PTL} \star)^i\ \mathsf{Mod})$.
2. $L((\boldsymbol{A} \circ \boldsymbol{G}_{com} \circ)^i\ \boldsymbol{A})$ $= L((\mathsf{PTL} \star \mathsf{Mod} \star)^i\ \mathsf{PTL})$.
3. $L(\boldsymbol{A} \circ (\boldsymbol{G}_{com} \circ \boldsymbol{A} \circ)^i\ \boldsymbol{G}_{com})$ $= L(\mathsf{PTL} \star (\mathsf{Mod} \star \mathsf{PTL} \star)^i\ \mathsf{Mod})$.
4. $L(\boldsymbol{G}_{com} \circ (\boldsymbol{A} \circ \boldsymbol{G}_{com} \circ)^i\ \boldsymbol{A})$ $= L(\mathsf{Mod} \star (\mathsf{PTL} \star \mathsf{Mod} \star)^i\ \mathsf{PTL})$.

Proof. The proof is by induction on i. Consider $i = 0$. Then (1) is the statement of Lemma 3.8 once we recall that any abelian group is a direct product of cyclic groups, and that a direct product can only recognize languages in the boolean closure of the languages recognized by each factor. (2) is Theorem 2.1. Now note that $\mathsf{PTL} = \mathsf{atomic} \star \mathsf{PTL}$, where atomic stands for the class of atomic PTL+MOD formulae a, for $a \in A$. Hence (3) follows from Corollary 3.7 by setting $\Phi = \mathsf{PTL}$, $\Psi = \mathsf{Mod}$, $\Xi = \mathsf{atomic}$, and appealing to cases (1) and (2) argued already. Then (4) follows analogously from $\mathsf{PTL} = \mathsf{PTL} \star \mathsf{atomic}$.

For the inductive step, consider (1). $L((\mathbf{G}_{com} \circ \mathbf{A} \circ)^{i+1}\ \mathbf{G}_{com}) = L(\mathbf{V} \circ \mathbf{W})$ where $\mathbf{V} = (\mathbf{G}_{com} \circ \mathbf{A})^i \circ \mathbf{G}_{com}$ and $\mathbf{W} = \mathbf{A} \circ \mathbf{G}_{com}$. By induction, $L(\mathbf{V}) = L(\Phi)$ where $\Phi = (\mathsf{Mod} \star \mathsf{PTL} \star)^i\ \mathsf{Mod}$, and $L(\mathbf{W}) = L(\Psi)$ where $\Psi = \mathsf{PTL} \star \mathsf{Mod}$. Thus $\Psi = \mathsf{PTL} \star \Xi$ with $\Xi = \mathsf{Mod}$, and Corollary 3.7 implies $L(\mathbf{V} \circ \mathbf{W}) = L(\Phi \star \Psi)$. This concludes the inductive step for case (1), because substitution is associative on individual formulae, so that $\star$ is associative on classes of formulae. Cases (2), (3) and (4) are proved analogously. ■

Remark 3.11 Because the $\ominus$ (previous) operator commutes to the right of any other operator, a proof closely resembling the above shows that replacing every occurrence of $\mathbf{G}_{com}$ above by an occurrence of $\mathbf{G}_{sol}$ would simply require replacing every occurrence of Mod by $\mathsf{Mod} \star \mathsf{Mod} \star \ldots \star \mathsf{Mod}$.

Remark 3.12 Not all $\mathsf{Mod}_{j,q}$ operators are required to attain the expressivity of PTL+MOD. Known techniques allow expressing $\mathsf{Mod}_{j,q}\phi$ as a boolean combination of $\mathsf{Mod}_{j,p}\psi$ formulae, only using primes p dividing q, while preserving the depth of alternation of Mods and PTLs in our hierarchies.

4 Group Temporal Logic

Can temporal logic capture nonsolvable regular languages? In this section we sketch a PTL+MOD extension to do just that. Unlike for the solvable languages however, there is no natural intuitive operation similar to $\mathsf{Mod}_{j,q}$ which ultimately yields all the nonsolvable group languages. Rather, it is as though each nonsolvable group needs to be indivisibly added to the logic (but see Remark 4.5). A similar approach was used before, for example when Barrington, Immerman and Straubing [2] added monoidal quantifiers (i.e. a restricted form of Lindström quantifiers [8]) to first-order logic.

By *group temporal logic* (PTL+GROUP) we will mean the usual past temporal logic supplemented with *group* temporal operators $\Gamma_{g,G}$ for any finite group G and $g \in G$. The operator $\Gamma_{g,G}$ always binds $|G|-1$ formulae. To define its semantics, we fix an ordering $g_1, g_2, \ldots, g_q = id$ of the elements of G. This ordering depends on G and is fixed forever. Given $u \in \Sigma_A^+$ and the PTL+GROUP formulae $\phi_1, \phi_2, \ldots, \phi_{q-1}$, we define for $0 \leq i' < |u|$ an element of G, denoted $\langle \phi_1, \phi_2, \ldots, \phi_{q-1} \rangle \langle u, i' \rangle$, as

$$\langle \phi_1, \phi_2, \ldots, \phi_{q-1} \rangle \langle u, i' \rangle = \begin{cases} g_1 & \text{if } (u,i') \models \phi_1, \\ g_2 & \text{if } (u,i') \models \neg\phi_1 \wedge \phi_2, \\ g_3 & \text{if } (u,i') \models \neg\phi_1 \wedge \neg\phi_2 \wedge \phi_3, \\ \ldots & \\ g_q & \text{if } (u,i') \models \neg\phi_1 \wedge \neg\phi_2 \wedge \ldots \wedge \neg\phi_{q-1} \end{cases}$$

and define $(u,i) \models \Gamma_{g,G}\langle \phi_1, \phi_2, \ldots, \phi_{q-1} \rangle$ to mean that $\prod_{i'=0}^{i} \langle \phi_1, \phi_2, \ldots, \phi_{q-1} \rangle \langle u, i' \rangle = g$.

Remark 4.1 The $\mathsf{Mod}_{j,q}$ operators of PTL+MOD were special cases of the group operators $\Gamma_{g,G}$. Indeed, when G is the cyclic group $(Z_q, +)$ and the elements $g_1, g_2, \ldots, g_q = id = 0$ of $(Z_q, +)$ are ordered so that $g_1 = 1$, we have $(u,i) \models \mathsf{Mod}_{j,q}\phi$ iff $(u,i) \models \Gamma_{j,Z_q}\langle \phi, \phi, \ldots, \phi \rangle$.

Substitution is defined as before, with the obvious new rule:

$$\begin{aligned} & \Gamma_{g,G}\langle \phi_1, \cdots, \phi_{|G|-1} \rangle [a \mapsto \psi_a] \\ = \ & \Gamma_{g,G}\langle \phi_1[a \mapsto \psi_a], \cdots, \phi_{|G|-1}[a \mapsto \psi_a] \rangle \end{aligned} \tag{7}$$

for every finite group G, $g \in G$, and any $|G|-1$ PTL+GROUP formulae $\phi_1, \cdots, \phi_{|G|-1}$. Associativity of substitution (Proposition 3.1) is preserved, and the inductive step in proving the substitution lemma (Lemma 3.2) for the case $\phi = \Gamma_{g,G}\langle \phi'_1, \ldots \phi'_{|G|-1} \rangle$ now becomes:

$$\begin{aligned} & (u,i) \models \phi[a \mapsto \psi_a] \\ \text{iff} \quad & \Pi_{i'=0}^{i} \langle \phi'_1[a \mapsto \psi_a], \ldots, \phi'_{|G|-1}[a \mapsto \psi_a] \rangle \langle u, i' \rangle \\ & = g \\ \text{iff} \quad & \Pi_{i'=0}^{i} \langle \phi'_1, \ldots, \phi'_{|G|-1} \rangle \langle u[a \mapsto \psi_a], i' \rangle = g \\ & \text{(induction hypothesis)} \\ \text{iff} \quad & (u[a \mapsto \psi_a], i) \models \Gamma_{g,G}\langle \phi'_1, \ldots \phi'_{|G|-1} \rangle \\ & \text{(semantics of } \Gamma_{g,G}). \end{aligned}$$

Lemma 3.3 extends as well; the inductive case $\phi = \Gamma_{g,G}\langle \phi'_1, \ldots \phi'_{|G|-1} \rangle$ becomes:

$$\begin{aligned} & u \in L(\phi[a \mapsto \psi_a]) \\ \text{iff} \quad & \Pi_{i=0}^{|u|-1} \langle \phi'_1[a \mapsto \psi_a], \ldots, \phi'_{|G|-1}[a \to \psi_a] \rangle \langle u, i \rangle \\ & = g \\ \text{iff} \quad & \Pi_{i=0}^{|u\sigma|-1} \langle \phi'_1, \ldots, \phi'_{|G|-1} \rangle \langle u\sigma, i \rangle = g \\ \text{iff} \quad & u\sigma \models \Gamma_{g,G}\langle \phi'_1, \ldots \phi'_{|G|-1} \rangle \\ \text{iff} \quad & u \in (L(\phi))\sigma^{-1}, \end{aligned}$$

where the second step uses the inductive hypothesis and the fact that $(u[0,i])\sigma = (u\sigma)[0,i]$ since σ is length-preserving.

Again here, in the general case of PTL+GROUP, lemmas 3.2, 3.3, 3.5 and 3.4 imply Theorem 3.6 exactly as argued in [14]. Our corollary 3.7 thus also holds.

Now recall Bool from Section 3.3 and let Group be the class of PTL+GROUP formulae with no operators other than the $\Gamma_{g,G}$ operators applied to Bool formulae.

Lemma 4.2 *Let G be a finite group.*

1. *If $g \in G$ and $\phi_1, \ldots, \phi_{|G|-1}$ from Bool, then G recognizes $L(\Gamma_{g,G}\langle \phi_1, \phi_2, \ldots, \phi_{|G|-1} \rangle)$.*
2. *If L is recognized by G, then L is a boolean combination of languages of the form $L(\Gamma_{g,G}\langle \phi_1, \phi_2, \ldots, \phi_{|G|-1} \rangle)$, where the ϕ_j are Bool formulae.*

Proof. Let $g_1, g_2, \ldots, g_{|G|} = id$ be the fixed ordering of G. To prove (1), define the morphism $\tau : \Sigma_A^+ \to G$ by setting for each $s \in \Sigma_A$

$$s\tau = \begin{cases} id & \text{if } (s,0) \models \bigwedge_{i=1}^{|G|-1} \neg\phi_i, \\ g_{\min\{k : (s,0) \models \phi_k\}} & \text{otherwise.} \end{cases}$$

Then, for any $u \in \Sigma_A^+$ and $0 \leq i < |u|$, $\langle \phi_1, \phi_2, \ldots, \phi_{|G|-1} \rangle \langle u, i \rangle = (u[i,i])\tau$. Hence $g\tau^{-1} = L(\Gamma_{g,G}\langle \phi_1, \phi_2, \ldots, \phi_{|G|-1} \rangle)$, which is thus recognized by G.

To prove (2), let $\tau : \Sigma_A^+ \to G$ recognize L. It suffices to express $g\tau^{-1}$ for any $g \in G$. Recall $\mathrm{iam}(s)$ from (5) and define, for each $g \in G$, the formula

$$\mathrm{iammappedto}(g) = \bigvee_{s \in g\tau^{-1}} \mathrm{iam}(s),$$

where by convention an empty disjunction is FALSE. Then, for any $u \in \Sigma_A^+$,

$$\begin{aligned} u \in g\tau^{-1} \quad \text{iff} \quad & u \models \Gamma_{g,G}\langle \mathrm{iammappedto}(g_1), \ldots, \\ & \mathrm{iammappedto}(g_{|G|-1}) \rangle. \end{aligned}$$

Theorem 4.3 *A language $L \subseteq \Sigma_A^+$ is expressible in PTL+GROUP if and only if L is regular.*

By the Krohn-Rhodes theorem, any finite semigroup belongs to a pseudovariety defined as the alternating iterated wreath product of $\mathbf{G}$ and $\mathbf{A}$. Hence Theorem 4.3 follows from Lemma 4.4, which is implied by Lemma 4.2 and Corollary 3.7 exactly as Lemma 3.10 was proved in the last section.

Lemma 4.4 *1.* $L((\boldsymbol{G} \circ \boldsymbol{A} \circ)^i \ \boldsymbol{G})$
$= L((\mathsf{Group} \star \ \mathsf{PTL} \star)^i \ \mathsf{Group})$.

2. $L((\boldsymbol{A} \circ \boldsymbol{G} \circ)^i \ \boldsymbol{A})$
$= L((\mathsf{PTL} \star \ \mathsf{Group} \star)^i \ \mathsf{PTL})$.

3. $L(\boldsymbol{A} \circ \ (\boldsymbol{G} \circ \ \boldsymbol{A} \circ)^i \ \boldsymbol{G})$
$= L(\mathsf{PTL} \star \ (\mathsf{Group} \star \ \mathsf{PTL} \star)^i \ \mathsf{Group})$.

4. $L(\boldsymbol{G} \circ \ (\boldsymbol{A} \circ \ \boldsymbol{G} \circ)^i \ \boldsymbol{A})$
$= L(\mathsf{Group} \star \ (\mathsf{PTL} \star \ \mathsf{Group} \star)^i \ \mathsf{PTL})$.

Remark 4.5 Do we need all nonsolvable group operators in order to capture all regular languages? Of course not. By the Krohn-Rhodes theorem, operators for the finite simple groups suffice. This remark in fact also applies to PLT+MOD: the $\mathsf{Mod}_{j,p}$ operators for prime p, from Remark 3.12, are precisely the abelian finite simple group operators, and wreathing the latter groups together generates all solvable groups.

5 Conclusion

Many extensions of temporal logic have been studied in the past. We proposed yet new extensions, with an eye towards an algebraic characterization of the languages defined. These extensions include, naturally, counters mod q, and more artificially, general group operators. Our characterization of PTL+MOD implies decidability of the logic. An open question is whether the levels of the natural PTL+MOD hierarchies induced are also decidable. Answering this question, even for the "since" hierarchy of PTL taken by itself, was already quite a challenge [14]. Yet deciding the levels of the various PTL+MOD hierarchies will likely require algebraic tools of comparable sophistication.

6 Appendix

This Appendix is intended for researchers unfamiliar with the sometimes intimidating algebraic techniques underlying our results. Its purpose is to provide some amount of intuition into our characterizations. The Appendix begins with a brief "primer" on the wreath product operation, which complements the description of the wreath product given in Section 2. In the full paper, this Appendix will include examples illustrating the correspondance between specific formulas and the automata and semigroups recognizing their languages.

6.1 A primer on wreath products

There are three related viewpoints on the wreath product operation. In Section 2, the most abstract viewpoint was chosen, and the wreath product was defined for abstract monoids. The abstract viewpoint is the cleanest and most appropriate for the purpose of analysis, but it provides little intuition.

A more concrete viewpoint on the wreath product is obtained by working with transformation monoids. A transformation monoid is just a monoid concretely represented as a set of transformations of a set Q (finite in our case). (A transformation is just a map from Q to itself, and the monoid operation is composition.) The transformation monoid viewpoint is appropriate for the purpose of synthesis, ie for being able to manipulate concise representations of monoids (on small sets Q, for example, or by restricting one's attention to subsets of a monoid generating the full monoid under composition).

A third viewpoint on the wreath product is most appropriate for gaining intuition (this "analysis-synthesis-intuition" nomenclature is that of Howard Straubing). This third viewpoint is obtained by combining finite automata sequentially in order to produce the effect of the wreath product. We now make this "cascade product" operation on finite automata precise. We will then use this operation to sketch how the regular languages in the variety $\mathbf{G}_{com} \circ \mathbf{A}$ are captured by formulas from $\mathsf{Mod} \star \mathsf{PTL}$.

A semi-automaton $(Q, A, \cdot)$ is a finite automaton in the usual sense, but with unspecified initial state and final states. (Here Q is a finite set of states, A is a finite alphabet, and $\cdot : Q \times A \to Q$ is its transition function.) Consider two semi-automata:

$$\begin{aligned} M_1 &= (Q, A, \cdot) \\ M_2 &= (P, Q \times A, \odot). \end{aligned}$$

Notice the alphabet of M_2. Intuitively, M_1 is a "front machine", which sees the input letters over the alphabet A as they arrive. The "back machine" M_2 sees both the input letters that arrive, *and* the current state of the front machine. The *cascade product* $M_2 \circ M_1$ is the semi-automaton obtained from connecting M_1 and M_2 in sequence. More precisely, $M_2 \circ M_1$ is the semi-automaton $(P \times Q, A, \otimes)$, where

$$\begin{aligned} \otimes : (P \times Q) \times A &\to P \times Q \\ ((p, q), a) &\mapsto (p \odot (q, a), q \cdot a). \end{aligned}$$

The intuitive gist of the wreath product operation is embodied in the cascade product of two automata. If we write $TM((Q, A, \cdot))$ for the monoid of transformations induced on the semi-automaton state set Q by all possible words over A, then the formal connection between the cascade product and the wreath product is the following: $TM(M_2 \circ M_1)$ *divides* the wreath product $TM(M_2) \circ TM(M_1)$. Here, division could be defined formally, but its intuitive meaning should be taken as "can be simulated by". In other words, the formal connection is that any regular language recognizable by the cascade product $M_2 \circ M_1$ can be "recognized" by the monoid $TM(M_2) \circ TM(M_1)$.

6.2 From wreath to formulas

Let $X = (Q, A, \cdot, s_0)$ be an automaton whose transformation semigroup is aperiodic and let $Y = (\{0, 1, \ldots, q-1\}, Q \times A, +, 0)$ be an automaton whose transformation semigroup is the cyclic group Z_q: note that this means that to every pair (t, a) in $Q \times A$ is associated an integer c, and the action of (t, a) in Y is to increase the value of the state by c mod q. Form the cascade product $Y \circ X$. We want to argue that, for any choice of accepting states in $Y \circ X$, the recognized language is expressible by a $\mathsf{Mod} \star \mathsf{PTL}$ formula. Since Mod includes the boolean connectives, it suffices to argue in the case where there is a unique accepting state $(c, s) \in \{0, 1, \cdots, q-1\} \times Q$.

When a word $x = a_1 \ldots a_n \in A^+$ is read by $Y \circ X$, the right-hand component of the state changes from s_0 to $s_0 \cdot x$, while the left-hand component changes from 0 to $(s_0, a_1) + (s_1, a_2) + \cdots (s_{n-1}, a_n)$ mod q, where $s_i = s_0 \cdot a_1 \ldots a_i$. Thus the word x is accepted iff $s_0 \cdot x = s$ and $(s_0, a_1) + (s_1, a_2) + \cdots (s_{n-1}, a_n) = c$ mod q.

The first condition is clearly expressible by a PTL formula since the transformation semigroup of X is aperiodic. The second condition can be expressed as a boolean combination of conditions of the form " there are d mod q occurrences of the pair (t, a) in the sum", for each (t, a) in $Q \times A$. Fix some pair (t, a): occurrences of this pair in the sum are in 1-1 correspondance with factorizations of x as $x = x_0 a x_1$, where $s_0 \cdot x_0 = t$. The number of occurrences is thus congruent to d mod q iff x satisfies the formula $\mathsf{Mod}_{d,q}(a \wedge \ominus \phi_t)$, where ϕ_t is the PTL formula describing the language $\{y : s_0 \cdot y = t\}$.

Because every abelian group is a direct product of cyclic groups, the above suffices to prove that any language recognized by a semigroup in the variety $\mathbf{G}_{com} \circ \mathbf{A}$ can be expressed by a $\mathsf{Mod} \star \mathsf{PTL}$ formula.

Bibliography

[1] J. Almeida, *Finite semigroups and universal algebra*, Series in Algebra, vol. 3, World Scientific, Singapore, 1995.

[2] D. A. Barrington, N. Immerman, H. Straubing, "On uniformity within NC^1," *J. Comput. System Sci.* **41**, 274-306 (1990).

[3] J. Brzozowski and R. Knast, The dot-depth hierarchy of star-free languages is infinite, *J. Computer and Systems Science* **16** (1978), pp. 37-55.

[4] J. Cohen, J.-E. Pin and D. Perrin, "On the expressive power of temporal logic", *Journal of Comput. Syst. Sci.* **46** (1993) 271-294.

[5] K. Etessami and Th. Wilke, "An until hierarchy for temporal logic". *LICS'96*.

[6] J.A. Kamp, "Tense Logic and the Theory of Linear Order", Ph.D. thesis, University of California, Los Angeles, 1968.

[7] K. Krohn and J. Rhodes, "The algebraic theory of machines I", *Trans. Amer. Math. Soc* **116** (1965) 450-464.

[8] P. Lindström. First order predicate logic with generalized quantifiers. *Theoria*, 32:186–195, 1966.

[9] R. McNaughton and S. Papert, *Counter-free Automata*, MIT Press, Cambridge, MA, 1971.

[10] M. Schützenberger, "On Finite Monoids Having Only Trivial Subgroups", *Information and Control*, **8** (1965) 190-194.

[11] H. Straubing, *Finite Automata, Formal Logic and Circuit Complexity*, Birkäuser, 1994.

[12] H. Straubing, D. Thérien and W. Thomas, "Regular languages defined with generalized quantifiers", in Proc. 15th ICALP, Springer Lecture Notes in Computer Sci., **317** (1988) 561-575.

[13] D. Thérien, T. Wilke, "Temporal logic and semidirect products: An effective characterization of the Until hierarchy". *FOCS'96*.

[14] D. Thérien, T. Wilke, "Temporal logic and semidirect products: An effective characterization of the Until hierarchy", Technical report 96-28, DIMACS, (1996) (URL: ftp://dimacs.rutgers.edu/pub/dimacs/TechnicalReports/TechReports/1996/ 96.28.ps.gz).

[15] W. Thomas, Classifying regular events in symbolic logic, *J. Computer and Systems Science* **25** (1982), pp. 360–376.