Around dot depth two

Kamal Lodaya[1], Paritosh K. Pandya[2], and Simoni S. Shah[2]

[1] The Institute of Mathematical Sciences, CIT Campus, Chennai *600113*, India
[2] Tata Institute of Fundamental Research, Colaba, Mumbai *400005*, India

Abstract. It is known that the languages definable by formulae of the logics $FO^2[<,S]$, $\Delta_2[<,S]$, $LTL[\mathsf{F},\mathsf{P},\mathsf{X},\mathsf{Y}]$ are exactly the variety $DA*D$. Automata for this class are not known, nor is its precise placement within the dot-depth hierarchy of starfree languages. It is easy to argue that $\Delta_2[<,S]$ is included in $\Delta_3[<]$; in this paper we show that it is incomparable with $\mathcal{B}(\Sigma_2)[<]$, the boolean combination of $\Sigma_2[<]$ formulae. Using ideas from Straubing's "delay theorem", we extend our earlier work [LPS08] to propose partially-ordered two-way deterministic finite automata with look-around (*po2dla*) and a new interval temporal logic called LITL and show that they also characterize the variety $DA*D$. We give effective reductions from LITL to equivalent *po2dla* and from *po2dla* to equivalent $FO^2[<,S]$. The *po2dla* automata admit efficient operations of boolean closure and the language non-emptiness of *po2dla* is NP-complete. Using this, we show that satisfiability of LITL remains NP-complete assuming a fixed look-around length. (Recall that for $LTL[\mathsf{F},\mathsf{X}]$, it is PSPACE-hard.)

A rich set of correspondences has been worked out between diverse mechanisms for defining the first-order definable word languages and their subclasses (a recent survey is [DGK08]). In the following, CFA refers to counter-free automata, SFRE to star-free regular expressions and *Ap* refers to the variety of aperiodic monoids [Pin86].

$$CFA \equiv SFRE \equiv Ap \equiv FO[<] \equiv LTL[\mathsf{U},\mathsf{S}] \equiv ITL$$

Further, Thomas showed [Tho82] that by restricting the quantifier-alternation depth in the $FO[<]$ formulae a strict dot-depth hierarchy of star-free languages is obtained, see the paper by Pin and Weil [PW97] for details. For example, $\mathcal{B}(\Sigma_2)[<]$ is the class of languages defined by the boolean combination of $\Sigma_2[<]$ formulae, which are the ones which have one block of existential quantifiers followed by one block of universal quantifiers followed by a quantifierless formula.

For the *FO* formulations below, given an alphabet A and $a \in A$, the unary predicate $Q_a(x)$ holds iff the letter at position x is a. The binary predicate $S(x,y)$ denotes the successor relation on positions, and $<$ is, as usual, its transitive closure.

Example 1. Let $A = \{a,b\}$ be the alphabet described by $\phi_A \stackrel{\text{def}}{=} \forall x.\ Q_a(x) \vee Q_b(x)$, which will be an additional conjunct below, not explicitly mentioned.

- $\phi_1 \stackrel{\text{def}}{=} \exists x \exists y.\ S(x,y) \wedge Q_a(x) \wedge Q_a(y)$ is a $\mathcal{B}(\Sigma_1)[S]$ formula defining $L_1 = A^*aaA^*$.
- $\phi_2 \stackrel{\text{def}}{=} \exists x \exists y.\ Q_a(x) \wedge Q_a(y) \wedge \forall z.\ (x < z \supset y \leq z)$ is a $\Sigma_2[<]$ formula defining L_1.

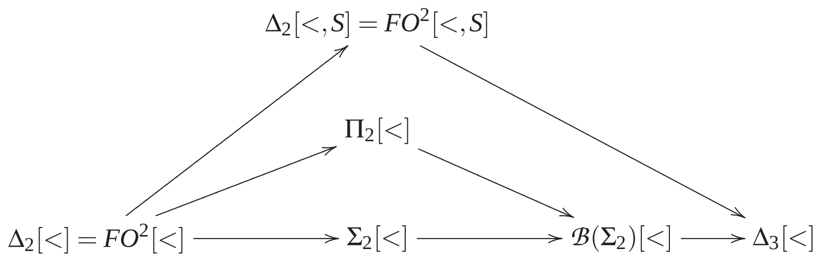
– Let $\phi_3 \stackrel{\text{def}}{=} (\forall x.\ first(x) \supset Q_a(x)) \wedge (\forall x.\ last(x) \supset Q_b(x)) \wedge$.
$(\forall x,y.\ ((x < y) \wedge Q_a(x) \wedge Q_a(y) \supset \exists z.\ x < z \wedge z < y \wedge Q_b(z))) \wedge$
$(\forall x,y.\ ((x < y) \wedge Q_b(x) \wedge Q_b(y) \supset \exists z.\ x < z \wedge z < y \wedge Q_a(z)))$
Then, ϕ_3 is a $\Pi_2[<]$ formula defining the language $L_2 = (ab)^*$. □

More recently, Thérien and Wilke [TW98] showed that the 2-variable fragment $FO^2[<]$ [Mor75] (where only two variables occur, quantified any number of times), is expressively equivalent to the unambiguous languages and variety DA of Schützenberger [Sch76,TT02] and the subset $\Delta_2[<]$ in the dot-depth hierarchy. Etessami, Vardi and Wilke [EVW02] identified the unary temporal logic $LTL[\mathsf{F},\mathsf{P}]$ and Schwentick, Thérien and Vollmer [STV02] identified partially-ordered 2-way deterministic finite automata (these are also called linear [LT00]) as equivalent formalisms. In [LPS08], we added to these correspondences a "deterministic" interval temporal logic called *UITL*. The papers [TW98,EVW02] also characterized $FO^2[<,S]$, which can define languages not definable in the logic $FO^2[<]$ such as those in Example 1. For a detailed study of these logics, see the recent papers of Weis and Immerman [WI07], and of Kufleitner and Weil [KW09].

$$PO2DFA \equiv UL \equiv DA \equiv FO^2[<] \equiv \Delta_2[<] \equiv LTL[\mathsf{F},\mathsf{P}] \equiv UITL$$

$$DA * D \equiv FO^2[<,S] \equiv \Delta_2[<,S] \equiv LTL[\mathsf{F},\mathsf{P},\mathsf{X},\mathsf{Y}]$$

It is clear that $\Delta_2[<,S] \subseteq \Delta_3[<]$ since successor can be defined using $<$ and one quantifier. In this paper we provide an automaton characterization and an interval logic characterization for this class of languages, and we separate it from $\mathcal{B}(\Sigma_2)[<]$, the languages defined by the boolean combination of $\Sigma_2[<]$ formulae. This also shows that $FO^2[<,S]$ is a *proper* subset of $\Delta_3[<]$, as diagrammatically depicted below.



Our automaton and logic characterizations are based on Rhodes expansions [Til76]; the two-sided variant below is inspired by Straubing's theorem $DA * D \equiv DA * LI$ [Str85].

Definition 1. *Let A be a finite alphabet, $A' = A \cup \{\triangleright, \triangleleft\}$ be its extension with two endmarkers $\triangleright, \triangleleft \notin A$, and $A^\rho_d = (A')^{2d+1}$ the alphabet whose letters are actually words of length $2d+1$ over A. Let $w = w_1 w_2 \ldots w_n$ be a given word, where $w_i \in A$ is a letter. Let $around_d(w,i) = w_{i-d} \ldots w_i \ldots w_{i+d}$ denote the* two-sided d-lookaround string *at position i. Note that if the position i is near one of the endpoints then $around_d(w,i)$ is padded by repeating the endmarker at that end. We define the* ***Rhodes-Straubing d-expansion*** *of w (and for a language L pointwise) for $d \geq 1$ to be $w^\rho_d = u_1 u_2 \ldots u_n$, where each $u_i = around_d(w,i)$. This is a word over A^ρ_d. When $d = 0$ we let w^ρ_0 be w. For example, $(abcab)^\rho_2$ is $(\triangleright\triangleright abc)(\triangleright abca)(abcab)(bcab\triangleleft)(cab\triangleleft\triangleleft)$.* □

Straubing's delay theorem shows that a language, or in our context a formula ϕ of $FO^2[<,S]$, can be seen as a formula ϕ' of $FO^2[<]$ over a Rhodes-Straubing d-expansion where d is the number of occurrences of successor predicates in ϕ. Carrying this intuition to automata, we extend *po2dfa* to partially-ordered 2-way deterministic finite state automata with lookaround (*po2dla*) which essentially make transitions on the Rhodes-Straubing expansion of the word. We also extend our unambiguous interval logic *UITL* to an unambiguous interval logic with lookaround called LITL. With some amount of technical hacking, we are able to show that LITL and *po2dla* have the expressive power of $FO^2[<,S]$.

The resulting automata and interval logic have many interesting features. A significant property of *po2dla* is that the boolean operations (including complementation) can be done within *po2dla* with a linear blowup in size. Language emptiness of *po2dla* is NP-complete and inclusion between *po2dla* is CoNP-complete, assuming a fixed lookaround size k.

The logic LITL inherits the desirable properties of its ancestor *UITL* [LPS08]. It admits unique parsability of models and exploiting this we can provide an efficient PTIME reduction from LITL to *po2dla*. This immediately gives us a small model property for the logic. Moreover, given a formula of length n with alphabet size m and lookaround length k, we can show that the satisfiability problem is in nondeterministic time $O((m^k)\times n)$. Assuming fixed lookaround size k, satisfiability is NP-complete. By comparison, the satisfiability of the logic $LTL[\mathsf{F},\mathsf{X}]$ is PSPACE-hard, although an action-indexed version was shown NP-complete by Muscholl and Walukiewicz [MW05].

The rest of the paper is organized as follows: the next section defines our automata, Section 2 the logic and the reductions from logic to automata and from automata to $FO^2[<,S]$. Section 3 deals with expressiveness and finally brings us back from $FO^2[<,S]$ to our logic.

Acknowledgements. We would like to thank the DLT referees for their detailed and encouraging reports and Sheng Yu, the PC co-chair, for gracefully allowing us to recover from some errors in the submitted version. KL would like to thank Manfred Kufleitner, Howard Straubing and Pascal Weil for discussions about the status of the logic $FO^2[<,S]$ within the dot-depth hierarchy during a visit in May-June '07 to LaBRI, Bordeaux, under the Indo-French research network project (P2R) Modiste-Cover/Timed-Discoveri. Pascal also introduced us to Rhodes expansions [Til76].

1 Partially-ordered two-way DFA with look-around

Fix an alphabet A and its extension $A' = A \cup \{\triangleright, \triangleleft\}$ with two endmarkers $\triangleright, \triangleleft \notin A$. Given $w \in A^*$, let $dom(w) = \{1,\ldots,|w|\}$. In recognizing w, the two-way automaton actually scans the string $w' = \triangleright w \triangleleft$ with letters $\triangleright$ and $\triangleleft$ at positions 0 and $|w|+1$ respectively. Thus, $dom(w') = \{0,\ldots,|w|+1\}$.

Let $a \in A'$ and let $u, v \in A^*$. We shall consider **patterns** of the form $u\underline{a}v$ with an underlined distinguished position. Given a pattern $u\underline{a}v$ and a word w', a position $i \in dom(w')$ matches the pattern, denoted $(w'[*,i,*] = u\underline{a}v)$, if the letter in w' at position i is a and this is followed by the string v (forward lookaround) and also this a is preceded by the string u (backward lookaround). Formally, $(w'[*,i,*] = u\underline{a}v)$ iff $w'[i] = a$ and $\forall k \in dom(v).\ i+k \in dom(w') \wedge w'[i+k] = v[k]$ and $\forall k \in dom(u).\ i-k \in dom(w') \wedge w'[i-k] = u[k]$. (When clear from the context, $u\underline{a}v$ will be written as uav).

For a string u, let $Pre(u)$ and $Suf(u)$ be the set of all prefixes and suffixes (respectively) of u. Given two patterns $u_1\underline{a_1}v_1$ and $u_2\underline{a_2}v_2$, we say that they are overlapping iff (i) $a_1 = a_2$, (ii) $u_1 \in Suf(u_2)$ or $u_2 \in Suf(u_1)$, and (iii) $v_1 \in Pre(v_2)$ or $v_2 \in Pre(v_1)$.

1.1 Automaton Definition

Partially ordered two-way DFA were introduced by Schwentick, Thérien and Vollmer [STV02] to characterize the unambiguous languages. We present a generalization with forward and backward lookaround. The transitions of the automaton are labelled by patterns over the alphabet, instead of letters. There is a default else transition associated with each state which is taken if no other transition is applicable. This makes our automata total.

Definition 2. *A* ***partially ordered 2DFA (****po2dla****) with lookaround size*** *k over A' is a tuple $M = (Q, \leq, \delta, s, t, r)$ where $(Q, \leq)$ is a finite partial order of states with distinct start, accept and reject states s, t and r where r and t are the only minimal elements of the poset and s is the only maximal element. Let $\mathcal{P}$ be the set of all patterns with a maximum lookaround of size k, i.e. the set of all $u\underline{a}v$ such that $u, v \in A^*$, $a \in A'$ and $|u|, |v| \leq k$. The transition function δ has two types of transitions: the matching transitions form a partial function $\delta_m : (Q \setminus \{t, r\} \times \mathcal{P}) \rightarrow (Q \times \{L, R, X\})$ where the first component q' of $\delta_m(q, u)$ satisfies $q' < q$, and the default else transition is a total function $\delta_{else} : (Q \setminus \{t, r\}) \rightarrow (Q \times \{L, R\})$ where the first component q' satisfies $q' \leq q$.*

Further, for determinism we have that, for all $q \in Q$, and $u_1\underline{a_1}v_1, u_2\underline{a_2}v_2 \in \mathcal{P}$, if $\delta_m(q, u_1\underline{a_1}v_1) = q_1$ and $\delta_m(q, u_2\underline{a_2}v_2) = q_2$ such that $q_1 \neq q_2$, then $u_1\underline{a_1}v_1$ and $u_2\underline{a_2}v_2$ are not overlapping. To ensure that the head of the automaton does not "fall beyond" the end-markers, we have an additional syntactic condition:

$$\forall q \in Q \setminus \{t, r\}\ .\ \exists q', q'' \in Q.\ \delta_m(q, \triangleright) = (q', R) \text{ and } \delta_m(q'', \triangleleft) = (q'', L). \qquad \square$$

A configuration of automaton M running on word w' is a pair (q, p) with $q \in Q,\ p \in dom(w')$. The automaton in a configuration (q, p) takes the unique δ_m transition from q, whose label is matched at the position p. If such a transition does not exist, then the automaton takes the default transition δ_{else} where it must change position.

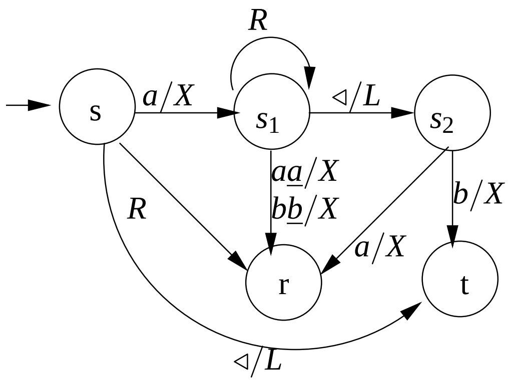
Run The *run* of the automaton M on a word w' and starting at a position p_0, is a sequence of state-position configurations $(q_0, p_0), (q_1, p_1)...(q_n, p_n)$, where

- $q_0 = s$ and $q_n \in \{t, r\}$. The run is accepting if $q_n = t$ and rejecting if $q_n = r$.
- For all $i \geq 0$, if there exists (unique) $u\underline{a}v$ such that $\delta_m(q_i, u\underline{a}v) = (q', d)$ for some (q', d) and $(w[*, i, *] = uav)$, then (a) $q_{i+1} = q'$ and (b) $p_{i+1} = p_i + 1$ if $d = R$, $p_{i+1} = p_i - 1$ if $d = L$ and $p_{i+1} = p_i$ if $d = X$.
- Otherwise, $q_{i+1} = q'$, where $\delta_{else}(q_i) = (q', d)$, and $p_{i+1} = p_i + 1$ if $d = R$ and $p_{i+1} = p_i - 1$ otherwise.

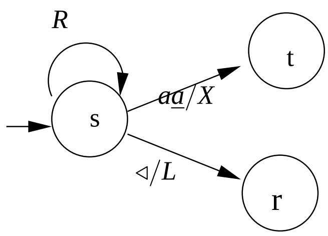
The outcome of the run is given by the total function $[[M]]$ such that for any $w \in A^*$ and $i \in dom(w')$ is given by $[[M]](w,i) = (q_n, p_n)$, the final configuration of the run. A word w is accepted by M if the unique run of M on $w' = \triangleright w \triangleleft$ starting at position 1 is accepting. The language $\mathcal{L}(M) \subseteq A^*$ is the set of words accepted by M. □

Since the states of M are partially ordered, the only loops allowed are self-loops on the default else transitions. During a sequence of such self-loop transitions the automaton moves in the same direction. Moreover, the automaton must change state on reaching an endmarker. So, because of the partial order, a *po2dla* cannot loop infinitely: it has at most $|Q|-1$ reversals and all its runs are bounded by length $|Q| \times |w|$. Since δ_{else} is a total function, the automaton always has a terminating run on every word: hence the automaton is complete.

Example 2. Figure 1 gives the *po2dla* for the languages $(ab)^*$ and A^*aaA^*. The default else transitions are shown with just a direction. In the automaton A_1, two consecutive a's or b's lead to rejection from state s_1, and in state s_2 which is reached at the end of the word, we check that it ends with b.



Automaton $\mathcal{A}_1$ accepting $(ab)^*$



Automaton $\mathcal{A}_2$ accepting A^*aaA^*

Fig. 1

Proposition 1. *The po2dla are closed under sequential composition and Boolean operations, constructible with automata of linear size (number of states).*

The proofs follow our earlier paper [LPS08]. Just as we have there, the automata can be described by a syntax of **extended turtle expressions** going beyond those of Schwentick, Thérien and Vollmer [STV02]. We omit these because of lack of space.

1.2 Small model property and decision problems

We let $INTV(w) = \{[i,j] \mid i,j \in dom(w), i \leq j\}$ be the set of intervals over w, and $w[i,j]$ (or $w,[i,j]$ in the next section) denote the factor of w corresponding to the interval $[i,j]$.

We will extend this notation to open and semi-open intervals as usual, as well as to their unions.

Consider a *po2dla* M over an alphabet A with n states and a maximum lookaround of k. Recall that $A' = A \cup \{\triangleright, \triangleleft\}$ and for $w \in A^*$ we have $w' = \triangleright w \triangleleft$. Recall also the defintion of $around_d(w,i)$ given in Definition 1. When clear from the context, we will abbreviate this by $around(w,i)$ or $around(i)$.

Lemma 1 (Membership). *Given a word $w \in A^*$, checking whether $w \in L(M)$ can be carried out by simulating the automaton in deterministic time $O(mnk)$ where m is the number of states of M, n is the length of the word w' and k is the lookaround size.*

Proof. Lookaround is handled by maintaining an array of length $2k+1$ storing the factor around the current head position. Note that there can be at most $m-1$ reversals in the *po2dla*. The algorithm requires space $\log m + \log n + (2k+1)\log |A'|$. □

Now consider the unique run of M accepting w. We say that a position $p \in dom(w')$ is purely-self-looping (PSL) if for all configurations of the form (q,p) in the run having position p, the (unique) enabled transition of M is the *self-looping else transition* (which does not result in change of state).

Call an interval $[m_1,m_2] \in INTV(w)$ a tunnel if all $j \in [m_1,m_2]$ are purely-self-looping (PSL) and $around(w',m_1) = around(w',m_2)$. If the automaton makes a right move at position m_1, it continues moving right without change of state till it reaches m_2; and similarly for a left move at m_2. The following lemma is a direct result of the above and the fact that $around(m_1) = around(m_2)$.

Lemma 2. *Given w' and a tunnel $[m_1,m_2]$, let $v' = w'[0,m_1)[m_2,|w'|]$ be the word obtained by replacing the tunnel by its last letter. Then, $w \in L(M)$ iff $v \in L(M)$.* □

From the above lemma, it is clear that every tunnel in word w' can be collapsed into a single letter preserving membership. Thus, in a word without tunnels, there can be a consecutive sequence of PSL positions which has length at most $|A'|^{2k+1}$ (the number of distinct $around(i)$). Every such sequence must be separated by a non-PSL position. There can be at most $n-1$ non-PSL positions in a run since there can be at most $n-1$ state changing transitions in an n state *po2dla*. Hence, we get the following theorem.

Theorem 1 (Small model). *If $L(M) \neq \emptyset$ then there exists a word $w \in L(M)$ of length at most $(|A'|^{2k+1}+1)(n-1)$.* □

Corollary 1. *Assuming lookaround k to be constant, the language non-emptiness of po2dla is NP-complete and the language inclusion of po2dla is CoNP-complete.*

Proof. The technique is to guess the member word of size $(|A'|^{2k+1}+1)(n-1)$ non-deterministically, and to use the PTIME membership checking algorithm on this. Thus, non-emptiness is in NP. The non-emptiness problem for *po2dfa* is shown to be NP-hard [SP09]. Since *po2dla* are extension of *po2dfa*, we conclude that their non-emptiness problem is NP-complete. We also conclude that the language inclusion problem is CoNP-complete as intersection and complementation of *po2dla* cause only linear blowup in the automaton size, and $L_1 \subseteq L_2$ iff $L_1 \cap \overline{L_2} = \emptyset$. □

2 Logic LITL

Interval temporal logic is based on a *chop* operator which divides an interval into two. Although this yields succinct formulae, the complexity of satisfiability is nonelementary. We proposed unambiguous interval temporal logic [LPS08] replacing chops by marked chop operators F_a and L_a, dividing a given interval at the first/last occurrence of the letter a. Satisfiability of *UITL* is NP-complete. Here we have a simple generalization, chopping an interval at the first/last occurrence of a given pattern $u\underline{a}v$.

Fix an alphabet A. Let $a \in A$ and $u, v \in A^*$. Let D, D_1, D_2 range over formulas in LITL. The abstract syntax of LITL is given below. Here $\top$ denotes the formula *true*.

$$\top \mid pt \mid D_1 \vee D_2 \mid \neg D \mid D_1 F_{u\underline{a}v} D_2 \mid D_1 L_{u\underline{a}v} D_2 \mid \oplus D \mid \ominus D$$

The satisfaction of a formula D is defined over intervals of a word model w as follows. As usual, $w \models D$ iff $w, [1, |w|] \models D$ and $L(D) \stackrel{\text{def}}{=} \{w \mid w \models D\}$ is the language defined by D. The derived operators $\wedge, \supset, \Leftrightarrow$ have their usual definitions.

$w, [i, j] \models pt$ iff $i = j$
$w, [i, j] \models D_1 F_{u\underline{a}v} D_2$ iff for some $k : k \in [i, j]$. $(w[*, k, *] = u\underline{a}v)$ and
for all $m : i \leq m < k$. $\neg(w[*, m, *] = u\underline{a}v)$ and
$w, [i, k] \models D_1$ and $w, [k, j] \models D_2$
$w, [i, j] \models D_1 L_{u\underline{a}v} D_2$ iff for some $k : k \in [i, j]$. $(w[*, k, *] = u\underline{a}v)$ and
for all $m : k < m \leq j$. $\neg(w[*, m, *] = u\underline{a}v)$ and
$w, [i, k] \models D_1$ and $w, [k, j] \models D_2$
$w, [i, j] \models \oplus D$ iff $i < j$ and $w, [i+1, j] \models D$
$w, [i, j] \models \ominus D$ iff $i < j$ and $w, [i, j-1] \models D$

Example 3. The LITL formula $\top F_{\underline{a}a} \top$ precisely specifies the language $A^* aa A^*$. The formula $(pt\ F_a \top) \wedge (\top L_b\ pt) \wedge \neg(\top F_{\underline{a}a} \top) \wedge \neg(\top F_{\underline{b}b} \top)$ specifies the language $(ab)^+$ over alphabet $\{a, b\}$. The first and the second conjunct state that the word begins with a and it ends with b. The last two conjuncts state that subwords aa or bb do not occur within the word.

2.1 Unique parsability and reduction to automata

As for its ancestor *UITL* [LPS08], every word model of a *LITL* formula can be uniquely parsed. Fix an LITL formula ϕ. Consider its subformula ψ occuring in context λ; we denote this by $\phi = \lambda(\psi)$. For any $w \in A^+$, we can uniquely determine if ψ is *relevant* in determining truth of ϕ over w. Moreover, if relevant, we can uniquely assign an interval $Intv_w(\psi)$ such that the truth value of ψ only over this interval is relevant in determining the truth of ϕ over w. The interval $Intv_w(\psi)$ actually depends only on the context λ and not on ψ. Moreover, it is possible to construct *po2dla* $\mathcal{L}(\psi)$ and $\mathcal{R}(\psi)$ which accept at the left and right interval boundaries of $Intv_w(\psi)$ respectively if the subformula is relevant. Using these automata, we can further construct an automaton $M(\psi)$ which accepts if ψ is relevant and it evaluates to true on $Intv_w(\psi)$. Exploiting this unique parsability, the following theorem can be established as a straightforward generalization of the similar theorem for logic *UITL* [LPS08].

Theorem 2. *For any $D \in LITL$ we can effectively construct a po2dla $M(D)$ in polynomial time such that $w \in \mathcal{L}(D) \Leftrightarrow w \in \mathcal{L}(M(D))$. The size $\mid M(D) \mid \; = \; O(\mid D \mid^2)$.*

Proof (sketch). The construction of $M(D)$ is inductive and proceeds bottom-up on the structure of D. Consider $D = \psi_1 F_{u\underline{a}v} \psi_2$. The corresponding *po2dla* $M(D)$ first moves to the left boundary of $Intv_w(D)$, then it checks in a single pass (moving in one direction only), for the existence of first $u\underline{a}v$, and also checks whether it lies within the right boundary of the interval $Intv_w(\psi)$. It then invokes the automata $M(\psi_1)$ and $M(\psi_2)$ in sequence. The details of the construction can be found in the full paper. □

Decision problems. The above translation gives a PTIME reduction from LITL formula of size n to a language equivalent automaton of size (i.e. number of states) $O(n^2)$. Moreover, the lookaround size in automaton is at most the pattern size in the LITL formula. Combining this with NP-complete non-emptiness checking of *po2dla*, we conclude that satisfiability of LITL is NP-complete assuming a fixed lookaround size. Our previous paper [LPS08] gave a LOGDCFL procedure for checking membership for logic *UITL*. This procedure extends to logic LITL with the same complexity.

2.2 From *po2dla* to $FO^2[<,S]$

In this section, we outline a language preserving translation from *po2dla* to $FO^2[<,S]$. Essentially the automaton is a dag with self-loops added on some nodes. For each progress edge $e = (p, \alpha, q, dir), p \neq q$, we define $FO^2[<,S]$ formulae $At_e(x)$ and $After_e(x)$ with one free variable x. These formulae satisfy the lemma below. By substituting these formulae as we go along the dag, we get a formula for the words accepted.

Lemma 3. – *$\triangleright w \triangleleft, i \models At_e(x)$ iff there exists a partial run of M (starting with $(s,1)$) which ends in configuration (p,i) and $(w[*,i,*] = \alpha)$.*
– *$\triangleright w \triangleleft, i \models After_e(x)$ iff there exists a partial run ending with last two configurations $(p,j)(q,i)$ where the last edge of the automaton taken is e.*

Construction. It is easy to construct $After_e(x)$ given $At_e(x)$. For edge $e = (p, \alpha, q, dir)$ we have $After_e(x) \stackrel{\text{def}}{=} \exists y.\, S(x,y) \wedge At_e(y)$ if $dir = L$; $After_e(x) \stackrel{\text{def}}{=} \exists y.\, S(y,x) \wedge At_e(y)$ if $dir = R$; and $After_e(x) \stackrel{\text{def}}{=} At_e(x)$ if $dir = X$.

Given α, there is a $FO^2[<,S]$ formula $\alpha(x)$ stating that the position x matches α. E.g. $da\underline{b}c(x) \stackrel{\text{def}}{=} b(x) \wedge (\exists y.\, S(y,x) \wedge a(y) \wedge (\exists x.\, S(x,y) \wedge d(x))) \;\wedge\; (\exists y.\, S(x,y) \wedge c(y))$.

Now we give the construction of $At_e(x)$, by induction on the depth of the edge. Consider an edge $e = (p, \alpha, q, dir)$ where the labels of other progress edges from state p are $\alpha_1, \ldots, \alpha_k$. Let the incoming progress edges to p be $e_1, \ldots, e_r$. We consider here the case that $\delta_{else}(p) = (p, R)$, i.e. a self-loop scanning rightwards. The case $\delta_{else}(p) = (p, L)$ is symmetric.

$$
\begin{aligned}
At_e(x) \stackrel{\text{def}}{=} \; & \alpha(x) \wedge \vee_{e_i \in \{e_1 \ldots e_r\}} [\\
& (\exists y.\, y \leq x \wedge After_{e_i}(y)) \wedge \\
& (\forall y.\, y < x \Rightarrow ((\neg\alpha(y) \wedge \neg\alpha_1(y) \wedge \ldots \wedge \neg\alpha_k(y)) \vee \\
& \qquad (\exists x.\, y < x \wedge After_{e_i}(x))))]
\end{aligned}
$$

For the start state s we assume that there is a dummy incoming edge e_{init} such that $After_{e_{init}}(x)$ is a formula which holds exactly at position 1 in w. The formula $\phi(M)$ for the whole automaton M is the disjunction of the formulae $At_{e_i}(x)$ for each incoming edge e_i to the accepting state t. Note that the size of $\phi(M)$ is exponential in size of M.

Theorem 3. *Every po2dla can be effectively reduced to a language equivalent formula of $FO^2[<,S]$ of exponential size.*

Hence, using Theorem 2, every LITL formula can be effectively reduced to a language equivalent $FO^2[<,S]$ formula, but a direct quadratic translation from LITL to $FO^2[<,S]$ generalizing the one in [Shah07] can also be worked out. In this paper, Theorem 5 will show that we can go from $FO^2[<,S]$ to LITL.

3 Games and expressiveness

We now investigate the expressiveness of $FO^2[<,S]$ with respect to the dot-depth hierarchy. Since a successor predicate can be replaced by $<$ with an additional nesting of a quantifier, we get that $FO^2[<,S] \subseteq \Delta_3[<]$.

Theorem 4. *(i)* $\Pi_2[<] \not\subseteq FO^2[<,S]$
(ii) $\Sigma_2[<] \not\subseteq FO^2[<,S]$
(iii) $FO^2[<,S] \not\subseteq \mathcal{B}(\Sigma_2)[<]$

To prove the above results, we use Ehrenfeucht-Fraïssé games [Fra50,Ehr61]. The signature has unary predicates Q_a, Q_b, Q_c and $<$ and S are the binary predicates, with their usual definitions. Let Sig be a signature, and u, v be two word structures over Sig. An EF game $G(u,v,p,r)$ is a game played by 2 players, the $Spoiler$ and $Duplicator$, over the word models u, v. A play of the game has r rounds with each player playing p pebbles. The pebbles are colored with p different colors, each player having exactly one pebble of each color.

In each round the $Spoiler$ picks (any) one of the words, and places his p pebbles on it. The $Duplicator$ then places his corresponding p pebbles on the other word. $Duplicator$ wins the game if at the end of r rounds there exists a partial isomorphism between the pebble positions, with respect to all the relations of Sig. Note that this can only happen if each of the intermediate configurations is also a partial isomorphism. Weis and Immerman [WI07] proved the following version of the Ehrenfeucht-Fraïssé theorem.

Definition 3. *Two words u, v are said to be r-$FO^2[<,S]$* equivalent *if for any $FO^2[<,S]$ formula ϕ with quantifier depth $\leq r$, $u \models \phi \Leftrightarrow v \models \phi$, and p-$\mathcal{B}(\Sigma_2)[<]$* equivalent *if for any $\mathcal{B}(\Sigma_2)[<]$ formula ϕ with $\leq p$ variables, $u \models \phi \Leftrightarrow v \models \phi$.*

Lemma 4. *(a) Two word models u, v over the signature $[<,S]$ are r-$FO^2[<,S]$ equivalent iff for the game $G(u,v,2,r)$, the $Duplicator$ always has a winning strategy.*
(b) Two word models u, v over the signature $[<]$ are p-$\mathcal{B}(\Sigma_2)[<]$ equivalent iff for the game $G(u,v,p,2)$, the $Duplicator$ always has a winning strategy.

Proof (of Theorem 4). We note that since $FO^2[<,S]$ is a boolean closed logic, (i) of the theorem will imply (ii) (or vice versa). We consider words over the alphabet $A = \{a,b,c\}$ described by a conjunct $\phi_A = \forall x(Q_a(x) \vee Q_b(x) \vee Q_c(x))$.
(i) We consider the language $(ac^*bc^*)^*$. This language may be expressed by the conjuncts below giving a $\Pi_2[<]$ formula:
$\forall x(\forall y(y < x \Rightarrow x = y)) \Rightarrow Q_a(x)$
$\forall x \exists y(Q_b(y) \wedge (x > y \Rightarrow Q_c(x)))$
$\forall x \forall y((Q_a(x) \wedge Q_a(y) \wedge x < y) \Rightarrow (\exists z.(x < z < y \wedge Q_b(z))))$ and
$\forall x \forall y((Q_b(x) \wedge Q_b(y) \wedge x < y) \Rightarrow (\exists z.(x < z < y \wedge Q_a(z))))$

For some $r > 0$, consider two word models over the signature $[<,S]$:
$u : (ac^r bc^r)^{2r}$, and $v : (ac^r bc^r)^r bc^r (ac^r bc^r)^r$
Here, $u \in (ac^*bc^*)^*$ and $v \notin (ac^*bc^*)^*$. We can show that for any 2-pebble, r-round EF game $G(u,v,2,r)$, the $Duplicator$ always has a winning strategy, and hence u,v are r-$FO^2[<,S]$ equivalent. This is evident from the observation that the two b's in v that do not have an a between them are separated by r c's and hence can be duplicated by the r^{th} bc^r in u. It is straightforward to see that any of the moves on a's or b's by the $Spoiler$ can be duplicated in the other word. So the language $(ac^*bc^*)^*$ cannot be expressed in $FO^2[<,S]$.
(iii) We show that the language given by the LITL formula $(\neg(\top F_{bb}\top))F_{aa}\top$ is not definable in $\mathcal{B}(\Sigma_2)[<]$. Over the signature $[<]$, we first claim that no formula using less than p variables can distinguish in one round between the words $u_1 = (ab)^p bb(ab)^p aa(ab)^p$ and $v_1 = (ab)^p aa(ab)^p bb(ab)^p$. This is because any subsequence of length p in one word can be matched in the other word.

Now consider the pair of words $u_2 = u_1^p$ and $v_2 = v_1^p$ formed by taking p copies of the earlier ones. Now any placement of p pebbles in one word can be matched in the other word so that the subwords of length at most $p-2$ between any two pebbles (or a pebble and an end of the word) are the same. This means that Duplicator has a winning strategy for the second round as well.

Since for every p, the first word u_2 is not in the language given by $(\neg(\top F_{bb}\top))F_{aa}\top$ and the second word v_2 is in the language, this shows that any $\mathcal{B}(\Sigma_2)[<]$ formula (using, say, p variables) fails to define the language. □

3.1 Using unambiguity on Rhodes-Straubing expansions

We now show that the expressiveness of $FO^2[<,S]$ is no more than that of LITL. Since the proofs of the lemmas are refinements of those in [TW98], they are omitted here. Let RS_d be the set of all words obtained as Rhodes-Straubing d-expansions (see Definition 1) of words over A, i.e. let $RS_d = (A^*)^\rho_d$. Our use of it is reminiscent of the rôle of Dyck languages in CFLs.

Lemma 5. *If a language L is defined by an $FO^2[<,S]$ sentence with at most r quantifier alternations and upto d successor formulas, then its d-expansion L^ρ_d is the intersection of RS_d with a language definable by a sentence of $FO^2[<]$ with at most r quantifier alternations.*

The letters occurring in a word x (more generally, in a set of words) are called its content. We will use $\|x\|$ to denote the size of the content of x. If the letter a is in $\|x\|$, the left a-chop of x is vaw where $x = v_1 a v_2$ and a is not in the content of v_1 (not in ithe content of v_2, respectively, for a right a-chop).

Definition 4 (Thérien and Wilke). *Let $n \geq \|x,y\|$. Any two words x and y are said to be $n,0$-equivalent. Two words x and y are $n,k+1$-*equivalent *if they have the same content and for every letter a in the content, if their left a-chops are $x_1 a x_2, y_1 a y_2$ respectively, then x_1 and y_1 are $n-1,k+1$-equivalent and x_2 and y_2 are n,k-equivalent, as well as a symmetric condition for right chops. The n,k-choppable languages are those which are a union of n,k-equivalence classes. The* ***unambiguously choppable*** *languages are those which are n,k-choppable for some $n,k > 0$.*

The next result combines the earlier proposition with the result of Thérien and Wilke [TW98] that an $FO^2[<]$-definable language is unambiguously choppable.

Corollary 2. *If a language L is defined by an $FO^2[<,S]$ sentence with upto d successor formulas, then its d-expansion L_d^ρ is the intersection of RS_d with an unambiguously choppable language.*

We now traverse the path back to our logic LITL.

Theorem 5. *The languages defined by sentences of $FO^2[<,S]$ can be defined in LITL.*

Proof. From Corollary 2, we know that for an $FO^2[<,S]$-definable language L (using d successors) over the alphabet A, its Rhodes-Straubing d-expansion $L^\rho \subseteq RS_d$ is unambiguously choppable over the alphabet $(A')^{2d+1}$. We construct LITL formulae for $\{w \mid w^\rho \in C^\rho\}$ and for each n,k-equivalence class $C^\rho \subseteq L^\rho$, by induction on n and k. Taking the disjunction of the formulae for the finitely many equivalence classes saturating L^ρ gives an LITL formula for L.

For the base case, an $n,0$-equivalence class determines a content B of letters over $(A')^{2d+1}$. The language recognized by words which map to this equivalence class is B^*, defined by the intersection below. Although B^* is a language over $(A')^{2d+1}$, the LITL formula is over the alphabet A since existence of a letter $u\underline{a}v$ in w^ρ is equivalent to validating $w \models trueF_{u\underline{a}v}true$.

- For lookarounds $u\underline{a}v \in (A')^{2d+1} \setminus B$ without padding, the conjunct is $\neg(trueF_{u\underline{a}v}true)$.
- For $\triangleright^i au\underline{b}v \in (A')^{2d+1} \setminus B$ where $i > 0$ and $|au|+i = d$, the conjunct is $\neg(ptF_{\underline{a}ubv}true)$.
- For $u\underline{a}vb\triangleleft^i \in (A')^{2d+1} \setminus B$ where $i > 0$ and $|vb|+i = d$, the conjunct is $\neg(trueL_{uav\underline{b}}pt)$.

For the induction step, an $n,k+1$-equivalence class determines a content B as well as a set of left and right α-chops for $\alpha \in B$. The required formula for an $n,k+1$-equivalence class is given by the following intersection, where in both cases we go through the endmarker analysis above and shift position as required.

- Formulae $D_1 F_\alpha D_2$ ($D'_1 L_\alpha D'_2$, respectively) and all allowed left (right, resp.) α-chops, where the lookarounds α range over the content.
- Negations of such formulae for the α-chops in $(A')^{2d+1}$ which are not allowed.

The formulae D_1, D_2, D'_1, D'_2 for n,k-classes over content B and for $n-1,k+1$-classes over content $B \setminus \{\alpha\}$ are obtained from the induction hypothesis. Consider for instance that $x^\rho = v^\rho \alpha w^\rho$ is a left α-chop over a Rhodes-Straubing expansion. The induction hypothesis gives us $v \models D_1$ and $w \models D_2$ and so we have $x \models D_1 F_\alpha D_2$ using α as a lookaround rather than a letter. □

References

[DGK08] V. Diekert, P. Gastin and M. Kufleitner. A survey on small fragments of first-order logic over finite words, *Int. J. Found. Comp. Sci.*, 19(3), 2008, 513–548.

[Ehr61] A. Ehrenfeucht. An application of games to the completeness problem for formalized theories, *Fund. Math.* 49, 1961, 129–141.

[EW00] K. Etessami and T. Wilke. An until hierarchy and other applications of an Ehrenfeucht-Fraïssé game for temporal logic, *Inf. Comput.* 160, 2000, 88–108.

[EVW02] K. Etessami, M.Y. Vardi and T. Wilke. First-order logic with two variables and unary temporal logic, *Inf. Comput.* 179, 2002, 279–295.

[Fra50] R. Fraïssé. Sur une nouvelle classification des systémes de relations, *Compt. Rend.* 230, 1950, 1022–1024.

[Imm98] N. Immerman. *Descriptive complexity* (Springer, 1998).

[KW09] M. Kufleitner and P. Weil. On FO^2 quantifier alternation over words, *Proc. 34th MFCS*, Novy Smokovec, (R. Královič and D. Niwiński, eds.), *LNCS* 5734 (Springer, 2009), 513–524.

[LPS08] K. Lodaya, P.K. Pandya and S.S. Shah. Marking the chops: an unambiguous temporal logic, *Proc. 5th IFIP TCS*, Milano (G. Ausiello, J. Karhumäki, G. Mauri and L. Ong, eds.), *IFIP Series* 273 (Springer, 2008), 461–476.

[LT00] C. Löding and W. Thomas. Alternating automata and logics over infinite words, *Proc. 3rd IFIP TCS*, Sendai (J. van Leeuwen, O. Watanabe, M. Hagiya, P.D. Mosses, T. Ito, eds.), *LNCS* 1872, 2000, 521–535.

[Mor75] M. Mortimer. On language with two variables, *Zeit. Math. Log. Grund. Math.* 21, 1975, 135–140.

[MW05] A. Muscholl and I. Walukiewicz. An NP-complete fragment of *LTL*, *Int. J. Found. Comp. Sci.* 16(4), 2005, 743–754.

[Pin86] J.-E. Pin. *Varieties of formal language* (North Oxford, 1986).

[PW97] J.-E. Pin and P. Weil. Polynomial closure and unambiguous products, *Theory Comput. Syst.* 30, 1997, 383–422.

[Sch76] M.-P. Schützenberger. Sur le produit de concaténation non ambigu, *Semigroup Forum* 13, 1976, 47–75.

[STV02] T. Schwentick, D. Thérien and H. Vollmer. Partially-ordered two-way automata: a new characterization of DA, *Proc. 5th DLT '01*, Vienna (W. Kuich, G. Rozenberg and A. Salomaa, eds.), *LNCS* 2295, 2002, 239–250.

[Shah07] S.S. Shah. FO^2 and related logics, Master's thesis (TIFR, 2007).

[SP09] S.S. Shah and P.K. Pandya, An automaton normal form for UITL, Technical Report STCS-TR-SP-2009/1 (Computer Science Group, TIFR, 2009).

[Str85] H. Straubing. Finite semigroup varieties of the form V*D, *J. Pure Appl. Algebra* 36(1), 1985, 53–94.

[TT02] P. Tesson and D. Thérien. Diamonds are forever: the variety DA, *Semigroups, algorithms, automata and languages* (G.M.S. Gomes, P.V. Silva and J.-E. Pin, eds.), (World Scientific, 2002), 475–500.

[TW98] D. Thérien and T. Wilke. Over words, two variables are as powerful as one quantifier alternation: $FO^2 = \Sigma_2 \cap \Pi_2$, *Proc. 30th STOC*, Dallas, 1998, 41–47.

[Tho82] W. Thomas. Classifying regular events in symbolic logic, *JCSS* 25(3), 1982, 360–376.

[Til76] B. Tilson. Complexity of semigroups and morphisms, Chapter XII of: S. Eilenberg, *Automata, languages and machines* B (Academic Press, 1976).

[WI07] P. Weis and N. Immerman. Structure theorem and strict alternation hierarchy for FO^2 on words, *Proc. CSL*, Lausanne (J. Duparc and T. Henzinger, eds.), *LNCS* 4646, 2007, 343–357.