Noname manuscript No.
(will be inserted by the editor)

Unary and two-variable interval logics

Kamal Lodaya

the date of receipt and acceptance should be inserted later

Abstract This paper shows that over finite word models, the interval logic of Halpern and Shoham (Halpern and Shoham (1991)) is expressively complete for two-variable logic with betweenness relations, introduced in Krebs et al. (2016). Satisfiability of formulae can be checked in polynomial space.

1 Introduction

Soon after Halpern et al. (1983); Moszkowski (1983) came up with interval temporal logic, with its characteristic binary *chop* operation, Halpern and Shoham (1991) defined unary interval logics, using modalities based on Allen (1983) interval algebra.

In these traditional interval logics (for example, see van Benthem (1983)), propositions are interpreted over intervals $[s,t]$, where s,t are positions in the word and $s \leq t$. That is, in the semantics they represent arbitrary binary relations over the domain (the set of positions of the word $\{1,2,\dots\}$). This plays a key role in Halpern and Shoham showing that satisfiability of their logic *HS* was undecidable over an infinite domain. Venema (1989) showed that the *HS* modalities can be represented in a plane, which allows encoding of grid problems. This result was sharpened, for example see Lodaya (2000); Bresolin et al. (2009b); Marcinkowski and Michaliszyn (2011). A large number of papers by Goranko, Montanari and others (see the papers Goranko et al. (2004); Bresolin et al. (2014)) have classified the decidability status of the satisfiability and model checking problems for nearly all fragments of the original Halpern-Shoham logic. *HS* is one of the best understood logics.

Realizing the difficulty of the two-dimensional interpretation, Halpern and Shoham suggested a different interpretation where propositions are interpreted at points, and hence their interpretation is a unary relation. Now the semantics of an interval logic formula is a first-order sentence over a linear order and hence

The Institute of Mathematical Sciences, CIT Campus, Chennai *600113*, and Homi Bhabha National Institute, Anushaktinagar, Mumbai *4000094*.

decidability is regained. This direction led to the Duration Calculus, an extension of interval logic with measurements studied by Zhou et al. (1991).

In this paper we only consider finite word models (the domain is $\{1, 2, \ldots, max\}$), over a finite alphabet (set of letters) Σ. Our tools will be automata and language theoretic. The discussion can be extended to timed words. In particular DC can be interpreted over timed words, providing a rich variety of features. The book by Zhou and Hansen (2004) gives many details of this work. Wilke (1994); Rabinovich (2000); Lodaya and Pandya (2006) define extensions to capture monadic second-order logic, since this remains decidable over word models (Thomas (1997) is a nice survey). Surprisingly validity and model checking of many practical examples can be successfully carried out, as demonstrated by Pandya (2001) using the Mona library of Basin and Klarlund (1995) for manipulating automata. Recently this work has been extended to synthesis for a rich variety of specifications by Wakankar et al. (2017).

Since first-order and monadic second-order logic over words have a mature algebraic theory of expressiveness (see the book of Straubing (1994)), over the last ten years we have applied it to interval logics. By the theorem of Kamp (1968) any first-order logic sentence over words can be expressed by repeatedly using at most three variables. In a couple of papers (Lodaya et al. (2008, 2010)), we characterized using interval logics two-variable logic over words $FO^2[<]$ and its extension $FO^2[<, Suc]$ with the successor relation. The characterization of the quantifier alternation hierarchy of $FO[<]$ (again see Thomas (1997)) remains largely open, except for a few lower levels. We review this work in Section 2.

Venema (1991) showed that Halpern-Shoham logics are not expressively complete for first-order logic. Intuitively they fall within the guarded fragment of Andréka et al. (1998), specialized to words. Bresolin et al. (2009a) showed that a fragment called neighbourhood logic is expressively complete for two-variable logic. They also showed that this is not true for HS.

Recently we algebraically characterized the logic $FO^2[<, bet]$, which extends two-variable logic with betweenness relations (Krebs et al. (2016, 2018)), alternately with counting upto a threshold, giving a larger fragment of first-order logic than two-variable logic. In this paper we attempt to tackle $FO^2[<, bet]$ using new fragments of Halpern-Shoham logic. In Section 3 we illustrate some key examples to show how they are formulated in interval logic. We are aided by a point temporal logic which we have already proved to be expressively complete in Krebs et al. (2016). In Section 4 we proceed in the other direction. Section 5 brings together all the results.

2 Two-variable logics and interval logics

In this paper we confine ourselves to models which are finite words, that is, discrete linear orders with domain $\{1, \ldots, max\}$ where each position has a letter (or colour) from a finite alphabet Σ. Thus we are in the context of formal language theory which is well-studied for decades (for example, see the book by Straubing (1994)).

A language is a set of words. For example, over the alphabet $\Sigma = \{a, b, c, d\}$, those words where, between the last a and subsequent first d, there must be no b. This is usually denoted by a regular expression $\Sigma^* a c^* d \{b, c, d\}^*$. That is, any

number of letters from Σ, then an a, then any number of c's, then a d, then any number of letters from b, c, d.

To avoid confusion, we do not use the word "language" to denote the formal setup of a logic, where we prefer to use "formulae" and "sentences". Our base logic will be two-variable first-order logic $FO^2[<]$ over words, with the linear order interpreting a binary predicate $<$ and letters at word positions interpreting unary predicates $a(.)$ for every $a \in \Sigma$. We will also consider $FO^2[<, Suc]$, which also has the successor partial function $y = x + 1$ interpreting a binary predicate $Suc(x, y)$. In Section 3 we will consider our richer logic $FO^2[<, bet]$.

Going further, one can use arbitrary numerical predicates for positions, for example $\exists y(x = 2 \times y)$ holds when position x has an even value. The language of even-length words and the language $\{a^n b^n \mid n \geq 1\}$ are defined using this kind of predicate. Regular expressions no longer suffice to describe such languages. Circuit families were proposed for this purpose, that is, the n'th circuit in the family describes the language L_n of words in a language L which are of length n. Since each L_n is a finite language, it can be defined by a boolean circuit (a directed acyclic graph with nodes, called "gates", labelled by boolean operations) with n inputs and one output for acceptance or rejection. The implicit restriction to the alphabet $\{0, 1\}$ can be modelled by coding bigger alphabets. It turns out Gurevich and Lewis (1984); Immerman (1987) that formulae of first-order logic on words using arbitrary numerical predicates correspond exactly to languages defined by families of circuits whose depth (length of longest path from an input to the output) is bounded by a constant.

2.1 Unambiguous interval temporal logic

In Lodaya et al. (2008), we defined an unambiguous interval logic. For example the language $\Sigma^* a c^* d \{b, c, d\}^*$ is defined by the formula

$$(true L_a((\neg(true F_b true)) F_d true)),$$

where L_a stands for the last a and F_b for the first b. More formally the logic has this syntax, where $a \in \Sigma$ is a letter and $P \subseteq \Sigma$ is a subset of letters:

$$\beta ::= pt \mid a \mid \lceil P \rceil \mid \lceil\lceil P \rceil \mid \lceil P \rceil\rceil \mid \lceil\lceil P \rceil\rceil \mid \beta_1 \vee \beta_2 \mid \neg\beta \mid \beta_1 F_a \beta_2 \mid \beta_1 L_a \beta_2 \mid \oplus\beta \mid \ominus\beta$$

Given two positions $s \leq t$ of a word w, let $[s, t]$ stand for the interval from position s to position t and $w[k]$ for the letter at position k.

$w, [s,t] \models pt$ iff $s = t$
$w, [s,t] \models a$ iff $s = t \wedge w[t] = a$
$w, [s,t] \models \lceil P \rceil$ iff for all $k : s < k < t : w[k] \in P$
$w, [s,t] \models \lceil\lceil P \rceil$ iff for all $k : s \leq k < t : w[k] \in P$
$w, [s,t] \models \lceil P \rceil\rceil$ iff for all $k : s < k \leq t : w[k] \in P$
$w, [s,t] \models \lceil\lceil P \rceil\rceil$ iff for all $k : s \leq k \leq t : w[k] \in P$
$w, [s,t] \models \beta_1 F_a \beta_2$ iff for some $k : s \leq k \leq t :$ $w[k] = a$ and
(for all $m : i \leq m < k : w[m] \neq a$) and
$w, [s,k] \models \beta_1$ and $w, [k,t] \models \beta_2$
$w, [s,t] \models \beta_1 L_a \beta_2$ iff for some $k : s \leq k \leq t :$ $w[k] = a$ and
(for all $m : k < m \leq j : w[m] \neq a$) and
$w, [s,k] \models \beta_1$ and $w, [k,t] \models \beta_2$
$w, [s,t] \models \oplus\beta$ iff $s < t$ and $w, [s+1,t] \models \beta$
$w, [s,t] \models \ominus\beta$ iff $s < t$ and $w, [s,t-1] \models \beta$

Our paper proved several results about this logic. First of all, even though it does not appear so from the semantics above, the logic has the two-variable property. We showed that all formulae of the logic have a unique parsing property and a translation to specially designed automata of Schwentick et al. (2002). Using their results, the translation extends to sentences of two-variable logic $FO^2[<]$. Using the automata, we showed that the complexity of model checking is in LogDCFL, and that of satisfiability is in Pspace (and NP over a fixed alphabet). Conversely, there is an exponential construction from $FO^2[<]$ to formulae of our logic which define the same language. A more general interval logic with the same expressiveness is reported in Lodaya and Pandya (2017).

Automata have a mature algebraic theory (see the books Pin (1986) and Straubing (1994)). Recall that a monoid is a set with an associative operation which has an identity element (usually denoted 1). The transition monoid of an automaton is effectively constructed by looking at all the transition functions from states to states realized by different words as inputs. The associative operation is function composition and the identity element is the identity function. Thus transitions performed by letters as well as words map to monoid elements. More precisely, for a language we use the transition monoid of the minimal automaton for the language, which is called its syntactic monoid. For the special automata of Schwentick et al. (2002), this transition monoid has nice properties, allowing an algebraic characterization of the languages in terms of an equational variety called **DA**, as also of two-variable logic from the work of Thérien and Wilke (1998), and thus of our unambiguous interval temporal logic. Tesson and Thérien (2002) is a detailed study of many properties of this variety.

We will briefly skim over this characterization. An idempotent element e in a monoid satisfies $e = ee$. Every loop in a minimal automaton (with words $l, ll, \dots$) will ultimately map to an idempotent. M_e is the submonoid generated by factors of e, that is, generated by $\{a \mid e = paq, \text{for some } p, q\}$. These are the words made up of letters which go into making up the loop. A monoid is in variety **DA** if for all its idempotents e, $eM_ee = \{e\}$. The equation for **DA** is simpler, but this condition is also checkable on a monoid. As a consequence of this characterization, whether a language is definable by a sentence of two-variable logic (equivalently, of our unambiguous logic) is checkable.

To give an example, suppose the loop $(ab)^*$ maps to an idempotent e. Then M_e is the language $\{a, b\}^*$. The definition of **DA** says that in this case, all of $\{a, b\}^*$

must map to that e. This gives a non-definability result: the language $(ab)^*$ does not map to a monoid in **DA**, since it cannot be distinguished from $\{a, b\}^*$.

A similar argument shows $\Sigma^* aa \Sigma^*$ is also not two-variable definable.

2.2 Lookaround interval temporal logic

In Lodaya et al. (2010), we considered an expanded version of our logic, where instead of modalities dealing with unambiguous occurrences of letters, we had modalities for unambiguous occurrences of substrings.

The formula $true F_{a\underline{a}} true$ says that there is an occurrence of the substring aa, where the underline indicates that the subintervals are positioned at the second a in the *first* occurrence of aa in the word. We saw that this language $\Sigma^* aa \Sigma^*$ is not two-variable definable. Thus this logic is more expressive. Since $(ab)^*$ is exactly those words which begin with a, end with b, and do not contain occurrences of the substrings aa and bb, it follows that this language is also definable in the lookaround logic by a longer formula.

The formula $\neg(true L_{b\underline{b}} true) F_{\underline{a}a} true$ says that the substring bb does not occur before the first occurrence of the substring aa.

Formally we have the following syntax, where $u \in \Sigma^+$ is a substring and $P \subseteq \Sigma^+$ is a finite subset of forbidden substrings of the form uav with letter $a \in \Sigma$ with prefix and suffix $u, v \in \Sigma^*$.

$$\beta ::= true \mid pt \mid u \mid \lceil \neg P \rceil \mid \lceil\lceil \neg P \rceil \mid \lceil \neg P \rceil\rceil \mid \lceil\lceil \neg P \rceil\rceil \mid$$
$$\beta_1 \vee \beta_2 \mid \neg\beta \mid \oplus \beta \mid \ominus \beta \mid \beta_1 F_{u\underline{a}v} \beta_2 \mid \beta_1 L_{u\underline{a}v} \beta_2$$

Given two positions $s \leq e$ of a word w, let $w[s..t]$ stand for the substring from position s to position t. The new modalities can be defined in this way, $\lceil \neg P \rceil, \lceil\lceil \neg P \rceil, \lceil\lceil \neg P \rceil\rceil$ being analogously extended from Section 2.1.

$w, [s,t] \models u$ iff $w[s..t] = u$

$w, [s,t] \models \lceil \neg P \rceil$ iff for all $k, m, u : s \leq k < m \leq t$ and $u \in P : w[k..m] \neq u$

$w, [s,t] \models \beta_1 F_{u\underline{a}v} \beta_2$ iff for some $k : s + |u| \leq k \leq t - |v| : w[k - |u|..k + |v|] = uav$
and (for all $m : s \leq m < k : w[m - |u|..m + |v|] \neq uav$)
and $w, [s,k] \models \beta_1$ and $w, [k,t] \models \beta_2$

$w, [s,t] \models \beta_1 L_{u\underline{a}v} \beta_2$ iff for some $k : s + |u| \leq k \leq t - |v| : w[k - |u|..k + |v|] = uav$
and (for all $m : k < m \leq t : w[m - |u|..m + |v|] \neq uav$)
and $w, [s,k] \models \beta_1$ and $w, [k,t] \models \beta_2$

Our results for the lookaround logic parallel those of the earlier paper. By following a proof analogous to our earlier one, we showed that the formulas of this logic translate to the two-variable logic $FO^2[<, Suc]$. For this purpose we had to define and work with more general (but still specialized) automata than those defined by Schwentick et al. (2002). The model checking problem for the logic is in LogDCFL, and the satisfiability problem is in Pspace. There is an exponential construction from an $FO^2[<, Suc]$ sentence to an equivalent formula in this logic defining the same language.

As in the case of our earlier logic, the logic $FO^2[<, Suc]$ has an algebraic characterization. Thérien and Wilke (1998) showed that it corresponds to the variety **LDA** of monoids M where, for every idempotent $e \neq 1$ of M, eMe is in **DA**. Now if we suppose that the language $(ab)^*$ maps to the idempotent e, this says that the language $(ab)^* \Sigma^* (ab)^*$, which is seen to be just the language Σ^* of all words,

should be definable in two-variable logic, which it is (by the sentence $true$). By our theorem, $(ab)^*$ is definable in $FO^2[<, Suc]$.

Again one can form non-definability arguments based on the algebraic characterization. Thus one shows first that the syntactic monoid for the language $c^*(ac^*bc^*)^*$ is not in **DA**. Then, since c maps to an idempotent (in fact, the identity), and $cMc = M$ is not in **DA**, the language is not in **LDA**. Hence it is not definable by a sentence of $FO^2[<, Suc]$.

2.3 Halpern-Shoham logics

The interval logic of Halpern and Shoham (1991) uses modalities corresponding to all the relations in the interval algebra of Allen (1983). We use some of these, B, E, D, A and $\overline{A}$, to model the logics we have seen above. All of these are subsumed by the *chop* modality of the original interval logic of Moszkowski (1983). Thus the syntax of the logic HS is as below, where $P \subseteq \Sigma$ is a finite subset of letters. Since we are dealing with a finite alphabet, we will freely use boolean formulas to represent P.

$\beta ::= pt \mid a \mid \lceil P \rceil \mid \lceil\lceil P \rceil \mid \lceil P \rceil\rceil \mid \lceil\lceil P \rceil\rceil \mid \langle \mathsf{B}pt\rangle\beta \mid \langle \mathsf{E}pt\rangle\beta \mid$
$\beta_1 \vee \beta_2 \mid \neg\beta \mid \langle \mathsf{B}\rangle\beta \mid \langle \mathsf{E}\rangle\beta \mid \langle \mathsf{D}\rangle\beta \mid \langle \mathsf{A}\rangle\beta \mid \langle \overline{\mathsf{A}}\rangle\beta$

One can write tricky formulas. For example, $\langle \mathsf{B}pt\rangle pt \wedge \lceil false \rceil \wedge \langle \mathsf{E}pt\rangle pt$ cannot be satisfied by a word of length more than two.

As usual in modal logic, there are derived box modalities, for example we have $[\mathsf{D}]\beta = \neg\langle \mathsf{D}\rangle\neg\beta$. As before, the semantics is defined on intervals $[s, t]$ of word w with $s \leq t$. Our during modality is strict on the left as well as the right.

$w, [s, t] \models a$ iff $s = t$ and $w[t] = a$
$w, [s, t] \models \langle \mathsf{B}pt\rangle\beta$ iff $w, [s, s] \models \beta$
$w, [s, t] \models \langle \mathsf{E}pt\rangle\beta$ iff $w, [t, t] \models \beta$
$w, [s, t] \models \langle \mathsf{B}\rangle\beta$ iff for some $m : s \leq m < t : w, [s, m] \models \beta$
$w, [s, t] \models \langle \mathsf{E}\rangle\beta$ iff for some $m : s < m \leq t : w, [m, t] \models \beta$
$w, [s, t] \models \langle \mathsf{D}\rangle\beta$ iff for some $m : s < k < m < t : w, [k, m] \models \beta$
$w, [s, t] \models \langle \mathsf{A}\rangle\beta$ iff for some $m : t \leq m : w, [t, m] \models \beta$
$w, [s, t] \models \langle \overline{\mathsf{A}}\rangle\beta$ iff for some $m : m \leq s : w, [m, s] \models \beta$

$\langle \mathsf{A}\rangle$ and $\langle \overline{\mathsf{A}}\rangle$ move from the current interval to a neighbouring interval on the right or left, respectively, while $\langle \mathsf{B}pt\rangle$ and $\langle \mathsf{E}pt\rangle$ shrink the current interval to a point. The fragment of HS with just these operators is sometimes called neighbourhood logic. Their semantics can be translated into $FO^2[<]$ by rotating variables. Bresolin et al. (2009a) showed that the converse holds and neighbourhood logic is expressively complete for two-variable logic $FO^2[<]$.

The unambiguous modalities $\beta_1 F_a \beta_2$ and $\beta_1 L_a \beta_2$ are modelled in HS as

$$\langle \mathsf{B}\rangle(\lceil\lceil \neg a \rceil \wedge \beta_1 \wedge \langle \mathsf{E}pt\rangle a \wedge \langle \mathsf{A}\rangle\beta_2), \quad \langle \mathsf{E}\rangle(\lceil \neg a \rceil\rceil \wedge \beta_2 \wedge \langle \mathsf{B}pt\rangle a \wedge \langle \overline{\mathsf{A}}\rangle\beta_1).$$

The first formula says that the current interval $[s, t]$ has a proper left subinterval $[s, m]$ with $m < t$ in which β_1 holds and an adjoining right subinterval after that in which β_2 holds. The second one is a mirror image. These HS formulae do not have identical semantics. Using the unique parsing property of our logic alluded to in Lodaya et al. (2008), the modelling does work correctly. Our results above shows that we can add the unambiguous B, E modalities to $A, \overline{A}$ (and even more, the

role of determinism is spelt out in Lodaya and Pandya (2017)), still maintaining the expressive completeness for two-variable logic.

Some patterns are going to appear frequently. We introduce derived modalities for invariance requirements on the B, D and E operators (without any unambiguity requirement).

Definition 1 Let $[\mathsf{D}P\lceil Q\rceil R]\beta = [\mathsf{D}](\langle \mathsf{B}pt\rangle P \wedge \lceil Q\rceil \wedge \langle \mathsf{E}pt\rangle R \supset \beta)$, and for the dual modality, $\langle \mathsf{D}P\lceil Q\rceil R\rangle\beta = \langle \mathsf{D}\rangle(\langle \mathsf{B}pt\rangle P \wedge \lceil Q\rceil \wedge \langle \mathsf{E}pt\rangle R \wedge \beta)$. If the P or R requirement is trivial, we drop it. If it is the same as the Q requirement, we use the closed interval form as an abbreviation. The fragment of *HS* logic, where only these derived forms of B, D and E are used with $P, Q, R \subseteq \Sigma$, is called *InvHS*.

Note that *InvHS* defines the same languages as *HS*, since we can use Σ for the sets P, Q, R if required.

The formulae $\beta_1 F_a \beta_2$ and $\beta_1 L_a \beta_2$ above can be written in *InvHS*:

$$\langle \mathsf{B}\lceil\neg a\rceil a\rangle(\beta_1 \wedge \langle \mathsf{A}\rangle\beta_2), \quad \langle \mathsf{E}a\lceil\neg a\rceil\rangle(\beta_2 \wedge \langle \overline{\mathsf{A}}\rangle\beta_1).$$

Let $u_1 \prec u_2$ and $u_1 \succ u_2$ denote that u_1 is a proper prefix, respectively suffix, of u_2. (Note that $\prec$ and $\succ$ are not converse to each other.) The lookaround modality $\beta_1 F_{u\underline{a}v}\beta_2$ is modelled in a formula below, which is not an *InvHS* formula, and which slightly extends *HS* by matching substrings against intervals rather than just letters:

$$\langle \mathsf{B}\rangle(\lceil\neg uav\rceil \wedge \beta_1 \wedge \langle \mathsf{E}\rangle ua \wedge \langle \mathsf{A}\rangle(\langle \mathsf{B}\rangle av \wedge \beta_2) \wedge \bigwedge_{u \succ u', uav = u'av'} \neg(\langle \mathsf{E}\rangle u'a \wedge \langle \mathsf{A}\rangle\langle \mathsf{B}\rangle av')).$$

The conjunctions at the end rule out the possibility of an earlier match of uav which starts at the end of the beginning subinterval. (Note that $v' \prec v$ is implied.) This will not be caught by forbidding the substring uav since both occurrences of $uav = u'av'$ stretch beyond the beginning subinterval. We do not introduce derived modalities here, we will not need them in the rest of the paper.

3 Interval logics with invariance

In our paper Krebs et al. (2016), we extended two-variable logic with binary predicates $a(x, y)$, $a \in \Sigma$, which say that $x < y$ and there is an occurrence of the letter a lying strictly between the two positions. The formula $\bigwedge_{a\in\Sigma} \neg a(x, y)$ says that there is no letter strictly between x and y, that is, that $y = x + 1$. Thus this logic subsumes $FO^2[<, Suc]$. There is a richer counting capability (up to a threshold) which can be modelled, we refer the reader to the paper.

In fact the logic extends further than $FO^2[<, Suc]$, since the language $c^*(ac^*bc^*)^*$ that we saw was not definable in that logic, can be defined by a sentence in the new logic. We also started on an algebraic characterization, which was completed in a second paper Krebs et al. (2018). Our result is that $FO^2[<, bet]$ corresponds to the variety $\mathbf{M_eDA}$ of monoids where, for every idempotent e in the monoid, the submonoid eM_ee is in $\mathbf{DA}$. Again this condition is effectively checkable on a monoid.

For example, in the syntactic monoid of the language $c^*(ac^*bc^*)^*$, c maps to an idempotent and the submonoid $cM_cc = \{c\}$ is the trivial singleton monoid, which is in **DA**. This holds for all idempotents and the syntactic monoid is in $\mathbf{M_eDA}$. A more involved argument from Krebs et al. (2016) uses the characterization to show that the regular language $(a(ab)^*b)^+$ is not definable in $FO^2[<, bet]$.

3.1 Modelling addition

We now attempt to design an interval logic which can describe betweenness features. Chandra et al. (1985) observed that the language $c^*(ac^*bc^*)^*$ is important in modelling integer addition. For simplicity let the base be 2 and work with the alphabet $\{0,1\}^3$. Let ADD be the language of words representing vertically three numbers $\binom{m}{m+n}$ written reversed (least significant bit first) in base 2.

For this to be correct, the i'th bit of sum is computed from the i'th bit of the inputs and whether or not there is a carry into bit i. If the i'th bit of both inputs is 1, we map the letter to a (we think of this as a *set* operation). If the i'th bit of both inputs is 0, we map to b (this is a *reset* operation). Otherwise we map to c (an *identity* operation).

Think of a monoid $(\{a,b,c\},\cdot,c)$ representing these three operations. The carry product is $x \cdot a = a$, $x \cdot b = b$ and $x \cdot c = x$ for an element x. The carry into bit $i+1$ of the sum is exactly when the first i monoid elements multiply to a. Now the carry computation of an addition corresponds to a word in the language $c^*(ac^*bc^*)^*$.

In *HS* logic we can say that there is a beginning subinterval ending with a and with possibly only c's preceding it, there is an ending interval beginning with b and possibly only c's following it, and there are no subintervals which begin and end with a and do not contain a b, nor those which begin and end with b and do not contain an a. That is, in *InvHS*:

$$\langle \mathsf{B}\lceil\lceil c\rceil a\rangle true \wedge \langle \mathsf{E}b\lceil c\rceil\rangle true \wedge \neg\langle \mathsf{D}a\lceil\neg b\rceil a\rangle true \wedge \neg\langle \mathsf{D}b\lceil\neg a\rceil b\rangle true.$$

This is also definable in $FO^2[<, bet]$. Hence the language ADD is also definable in *InvHS* and in $FO^2[<, bet]$.

3.2 Defining circuits

In Krebs et al. (2016) we showed that our logic $FO^2[<, bet]$ can define languages arbitrarily high in the quantifier alternation hierarchy of first-order logic on words (see Thomas (1997)) and the constant-depth circuit hierarchy of Sipser (1983). The entire latter hierarchy corresponds to first-order logic on words using arbitrary numerical predicates, as we said in Section 2. For simplicity assume that the circuits have levels consisting of gates of only one type, alternating between *or* and *and*. With alphabet $\{0, 1, o_1, a_2, o_3, \ldots, a_{2k}, o_{2k+1}, \ldots\}$ one can encode in prefix form constant-depth boolean circuits with inputs set to 0 and 1, upto any finite level. We will see examples below. In our encoding we use $\triangleleft$ as a right endmarker for the input word signifying the circuit and propositions $P_0, P_1, \ldots$ for letters from subalphabets, which we will define.

Our aim is to show that the logic *InvHS* can describe circuits of arbitrary depth and hence represent regular languages at any level of the constant-depth

hierarchy, using more and more such propositions. Using its semantics, it is thus a rich fragment of first-order logic $FO[<]$ on words. The formulation of Sipser (1983) is used since it is amenable for our logic.

Let $Circ_1 = o_1\{0,1\}^+$ be the regular language encoding circuits of depth one (an *or* gate followed by inputs). Let P_1 be the subalphabet $\{o_1, a_2, o_3, \ldots, \lhd\}$. The *InvHS* formula is

$$C_1 = \langle \mathsf{B} o_1 \lceil \neg o_1 \rceil P_1 \rangle \lceil 0 \vee 1 \rceil.$$

Further, circuits which evaluate to *true* need to have at least one 1, the language is $True_1 = o_1\{0,1\}^*1\{0,1\}^*$. The *InvHS* formula is

$$T_1 = \langle \mathsf{B} o_1 \lceil \neg o_1 \rceil P_1 \rangle (\lceil 0 \vee 1 \rceil \wedge \neg \lceil 0 \rceil).$$

The next level is the regular language $Circ_2 = a_2(Circ_1)^+$ (an *and* gate at level two, followed by depth one circuits). Let P_2 be the subalphabet $\{a_2, o_3, \ldots, \lhd\}$. The *InvHS* formula can be written in two ways. The second formula shows how the endmarker can be used to convert invariant during formulae to invariant prefix and suffix formulae.

$$C_2 = [\mathsf{D} a_2 \lceil \neg a_2 \rceil P_2] C_1, \text{ or } C_2' = [\mathsf{E} \lceil \neg \lhd \rceil P_2][\mathsf{B} a_2 \lceil \neg a_2 \rceil P_2] C_1.$$

$True_2 = a_2(True_1)^+$ describes the circuits evaluating to *true* at the second level. The *InvHS* logic formula forces every conjunct to have a true disjunct:

$$T_2 = [\mathsf{D} a_2 \lceil \neg a_2 \rceil P_2] T_1.$$

The third level is the regular language $Circ_3 = o_3(Circ_2)^+$ which is handled by defining $P_3 = \{o_3, \ldots, \lhd\}$ and

$$C_3 = [\mathsf{D} o_3 \lceil \neg o_3 \rceil P_3] C_2.$$

Now the circuits evaluating to *true* are described by $True_3 = o_3(Circ_2)^* True_1 (Circ_2)^*$. The *InvHS* formula requires one true disjunct among all the disjuncts:

$$T_3 = C_3 \wedge (\langle \mathsf{D} o_3 \lceil \neg o_3 \rceil P_3 \rangle T_2).$$

It should be clear how to extend this to circuits of any finite level. Thus, like $FO^2[<, bet]$, the logic *InvHS* can also define languages arbitrarily high in the quantifier alternation hierarchy.

3.3 Defining monomials

We now show that we can define languages arbitrarily high in the nested until/since hierarchy of linear temporal logic on finite words, worked out by Thérien and Wilke (2004) (see the survey by Tesson and Thérien (2007)). To do this we will show that one can use a series of between operations, briefly a *monomial* language $P_0^* a_1 P_1^* \ldots P_{n-1}^* a_n P_n^*$ for some subalphabets $P_0, P_1, \ldots, P_{n-1}, P_n$ (again interpreted by propositions), followed by a formula β. To do this in temporal logic one would require $n+1$ nested until operators.

The formula of *HS* logic is: $F_0 = \langle \mathsf{B} \rangle (F_1 \wedge \langle \mathsf{A} \rangle \beta)$, where the formulae $F_m,\ m \geq 1$, expand the monomial, starting with $F_1 = \langle \mathsf{B} \lceil P_0 \rceil \rangle \langle \mathsf{A} \rangle F_2$, $F_2 = \langle \mathsf{B} a_1 \lceil P_1 \rceil \rangle \langle \mathsf{A} \rangle F_3$, up

to $F_n = \langle \mathsf{B} a_{n-1} \lceil P_{n-1} \rceil \rangle \langle \mathsf{A} \rangle F_{n+1}$, ending with $F_{n+1} = \langle \mathsf{B} pt \rangle (a_n) \wedge \lceil P_n \rceil$. We can also allow, instead of a single monomial, a boolean collection of monomials before the satisfaction of β. This latter formula shows that the second level of the quantifier alternation hierarchy (see Thomas (1997)) is subsumed in *HS* logic.

Another way is to run the induction down from $n+1$ to describe the languages formed as one moves left in the monomial, to obtain an *InvHS* formula scheme. The base step is: $F'_{n+1} = \beta$. The induction step is: $F'_{m-1} = \langle \mathsf{B} a_{m_1} \lceil P_{m-1} \rceil \rangle \langle \mathsf{A} \rangle F'_m$. The final formula is: $F'_0 = \lceil\lceil P_0 \rceil \langle \mathsf{A} \rangle F'_1$. If β was in *InvHS*, so are the formulae F'_m. It follows that the logics *InvHS* and $FO^2[<, bet]$ can define languages arbitrarily high in the until/since hierarchy of temporal logic.

Our examples have given sufficient evidence that our interval logic with invariance *InvHS* rivals the two-variable logic $FO^2[<, bet]$ in expressiveness.

Lemma 2 *InvHS subsumes* $FO^2[<, bet]$.

Proof We rely on a theorem in Krebs et al. (2016), which shows that definability in the latter logic is achieved by a point temporal logic with modalities which, for our purposes here, we can read as:

$$(\neg R) \textit{ until } \alpha \text{ and } (\neg R) \textit{ since } \alpha, \text{ for } R \subseteq \Sigma.$$

But the translation from monomials to *InvHS* that we have given above is already a generalization of the first modality, and a mirrored translation will handle the second one. This gives us a direct linear translation from the point temporal logic, and hence a translation from $FO^2[<, bet]$, to *InvHS*. □

4 Staying within two-variable between logic

As we have seen so far, the Halpern-Shoham logic with invariance *InvHS* is expressive enough to encompass our earlier two-variable logic with betweenness from Krebs et al. (2016). In this section we see if we have gone too far and overshot this logic.

Definition 3 The fragment of *InvHS* logic, where the derived forms of the invariance requirements B and E are used only in the forms $\langle \mathsf{B} P \lceil Q \rceil R \rangle \beta$ and $\langle \mathsf{E} P \lceil Q \rceil R \rangle \beta$, where β is a boolean combination of $\langle \mathsf{A} \rangle$ formulas or a boolean combination of $\langle \overline{\mathsf{A}} \rangle$ formulas, is called *bundled InvHS* (to borrow some jargon from Padmanabha et al. (2018)). Assuming our words are provided with endmarkers, we also allow invariance D requirements.

Lemma 4 *Bundled InvHS is expressively complete for* $FO^2[<, bet]$.

Proof There is a standard linear translation from neighbourhood logic (with the $A, \overline{A}$ modalities) to two-variable logic $FO^2[<]$. From this it follows that prefixing the neighbourhood modalities with invariance requirements to form the modalities $\langle \mathsf{B} P \lceil Q \rceil R \rangle \langle \mathsf{A} \rangle \beta$ and $\langle \mathsf{E} P \lceil Q \rceil R \rangle \langle \overline{\mathsf{A}} \rangle \beta$ linearly translates into the extended logic $FO^2[<, bet]$, the invariance requirements were designed to do so. The same holds if one takes boolean combinations of the $\langle \mathsf{A} \rangle$ or the $\langle \overline{\mathsf{A}} \rangle$ formulas. Assuming endmarkers, it follows that the invariance-guarded $\langle \mathsf{D} \rangle$ modality can be eliminated as in Section 3.2 and the resulting formula translated into $FO^2[<, bet]$. Combining with Lemma 2, bundled *InvHS* equals the expressive power of two-variable between logic. □

We can push this idea further. From Section 2.3 we know that when unambiguously guarded, the formula $\langle \mathsf{B}P\lceil Q\rceil R\rangle(\beta_1 \wedge \langle \mathsf{A}\rangle\beta_2)$ which is syntactically not in the bundled fragment, is definable in $FO^2[<]$, hence also in $FO^2[<, bet]$. What happens when we drop the unambiguity requirement? In our earlier papers Lodaya et al. (2008, 2010) we used unique parsing properties and automaton characterizations to traverse the path from interval logic to two-variable logic. These are lacking for the logic *InvHS*. We obtain a positive result by translating *InvHS* to its bundled fragment.

Lemma 5 *InvHS is expressively complete for* $FO^2[<, bet]$.

Proof Given Lemmas 2 and 4, it is sufficient to take care of formulae which are outside the bundled fragment of *InvHS*. We take a representative type of such a formula, an invariance-guarded $\langle \mathsf{B}\rangle$ formula. The same approach works for other invariance-guarded formulae.

Consider for various cases of the subformula β_1, which we name for this proof a *loose* subformula, in an *InvHS* formula which is not inside an $\langle \mathsf{A}\rangle$ or $\langle \overline{\mathsf{A}}\rangle$ modality:

$$\langle \mathsf{B}P\lceil Q\rceil R\rangle(\beta_1 \wedge \langle \mathsf{A}\rangle\beta_2).$$

We will make use of the number of *direction alternations* in a formula: each occurrence of a $\langle \mathsf{D}\rangle$ or $\langle \mathsf{E}\rangle$ modality immediately inside a $\langle \mathsf{B}\rangle$ modality, and each occurrence of a $\langle \mathsf{D}\rangle$ or $\langle \mathsf{B}\rangle$ modality immediately inside a $\langle \mathsf{E}\rangle$ modality, and each occurrence of a $\langle \mathsf{B}\rangle$,$\langle \mathsf{E}\rangle$,$\langle \mathsf{D}\rangle$ modality immediately inside a $\langle \mathsf{D}\rangle$ modality, counts as a direction alternation.

In each case, we give rules to rewrite this formula to a formula "closer" to the bundled fragment of *InvHS*, that is, where each invariance-guarded modality is immediately followed by an $\langle \mathsf{A}\rangle$ or $\langle \overline{\mathsf{A}}\rangle$ modality. The rules have simple ideas: requirements are pulled out to the endpoints of the interval if possible, otherwise nested modalities are converted to a sequence of modalities "fenced" by $\langle \mathsf{A}\rangle$,$\langle \overline{\mathsf{A}}\rangle$ modalities. This reduces the modal depth of loose subformulae. This does not happen in case there is a direction alternation between adjoining modalities, the best the sequencing can do is to process them in the same direction, reducing the number of such alternations.

Let us run through the cases. If β_1 is a point formula, its point requirements can be handled in the invariance guard, with any $\langle \mathsf{A}\rangle$,$\langle \overline{\mathsf{A}}\rangle$ modal requirements being pulled out. Disjunctions can also be pulled out.

Now we have cases where β_1 is headed by a modality followed by a formula γ_1. If $\beta_1 = \langle \overline{\mathsf{A}}\rangle\gamma_1$ (or even a boolean combination of such formulas), it can be pulled out. Nested invariance-guarded B formulae first have an intersecting interval, then the remaining outer condition is fulfilled. We call this *distribution* of the invariance requirements. Nesting an invariance-guarded $[\mathsf{E}]$ formula is dealt with similarly, but a nested $\langle \mathsf{E}\rangle$ formula is linearized. Nesting an invariance-guarded $[\mathsf{D}]$ formula combines features from the treatment of nested $[\mathsf{B}]$ and $[\mathsf{E}]$ formulae, a nested $\langle \mathsf{D}\rangle$ formula combines features from the treatment of nested $\langle \mathsf{B}\rangle$ and $\langle \mathsf{E}\rangle$ formulae.

In each case either the formula mapped to is already in the bundled fragment, or the loose subformulae in the mapped formula have lesser modal depth than β_1, or the number of alternations in the mapped formula are lesser than β_1. Thus the rules can be recursively applied to obtain a reduction into the bundled fragment.

Here are the rewrite rules for these cases, the left column gives the loose formula β_1 being considered. We also specify the closeness achieved in each rule.

$\langle\overline{\mathsf{A}}\rangle\gamma_1 \wedge \neg\langle\overline{\mathsf{A}}\rangle\gamma_2$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle(\langle\overline{\mathsf{A}}\rangle\gamma_1 \wedge \neg\langle\overline{\mathsf{A}}\rangle\gamma_2 \wedge \langle\mathsf{A}\rangle\beta_2)$, *modal depth decreases* $\hookrightarrow \langle\overline{\mathsf{A}}\rangle\gamma_1 \wedge \neg\langle\overline{\mathsf{A}}\rangle\gamma_2 \wedge \langle \mathsf{B}P\lceil Q\rceil R\rangle\langle\mathsf{A}\rangle\beta_2$
$[\mathsf{B}P'\lceil Q'\rceil R']\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle([\mathsf{B}P'\lceil Q'\rceil R']\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *modal depth decreases* $\hookrightarrow [\mathsf{B}(P\wedge P')\lceil Q\wedge Q'\rceil R']\gamma_1 \wedge \langle \mathsf{E}pt\rangle(R \wedge \langle\mathsf{A}\rangle\beta_2)$
$\langle\mathsf{B}P'\lceil Q'\rceil R'\rangle\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle(\langle\mathsf{B}P'\lceil Q'\rceil R'\rangle\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *modal depth decreases* $\hookrightarrow \langle\mathsf{B}(P\wedge P')\lceil Q\wedge Q'\rceil(R\wedge R')\rangle(\gamma_1 \wedge \langle\mathsf{A}\rangle\langle\mathsf{E}R'\lceil Q\rceil R\rangle\beta_2)$
$[\mathsf{E}P'\lceil Q'\rceil R']\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle([\mathsf{E}P'\lceil Q'\rceil R']\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *modal depth decreases* $\hookrightarrow \langle \mathsf{B}pt\rangle P \wedge [\mathsf{E}P'\lceil Q\wedge Q'\rceil(R\wedge R')](\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$
$\langle\mathsf{E}P'\lceil Q'\rceil R'\rangle\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle(\langle\mathsf{E}P'\lceil Q'\rceil R'\rangle\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *direction alternation decreases* $\hookrightarrow \langle \mathsf{B}P\lceil Q\rceil P'\rangle\langle\mathsf{A}\rangle(\langle\mathsf{B}P'\lceil Q\wedge Q'\rceil(R\wedge R')\rangle\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$
$[\mathsf{D}P'\lceil Q'\rceil R']\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle([\mathsf{D}P'\lceil Q'\rceil R']\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *modal depth decreases* $\hookrightarrow \langle \mathsf{B}pt\rangle P \wedge [\mathsf{D}P'\lceil Q\wedge Q'\rceil R'](\gamma_1 \wedge \langle\mathsf{E}pt\rangle(R \wedge \langle\mathsf{A}\rangle\beta_2))$
$\langle\mathsf{D}P'\lceil Q'\rceil R'\rangle\gamma_1$	$\langle \mathsf{B}P\lceil Q\rceil R\rangle(\langle\mathsf{D}P'\lceil Q'\rceil R'\rangle\gamma_1 \wedge \langle\mathsf{A}\rangle\beta_2)$, *direction alternation decreases* $\hookrightarrow \langle \mathsf{B}P\lceil Q\rceil P'\rangle\langle\mathsf{A}\rangle(\langle\mathsf{B}P'\lceil Q\wedge Q'\rceil(R\wedge R')\rangle(\gamma_1 \wedge \langle\mathsf{A}\rangle\langle\mathsf{E}P\lceil Q\rceil R\rangle\beta_2))$

Finally consider the cases where β_1 is a boolean combination of invariance-guarded modalities. We use the fact that there is a linear ordering and there are only a few order types to deal with, which we can disjunct over. Two invariance-guarded B formulae in the same interval can be ordered, or seen as one of two possible orderings. If there is a B and an E formula, there are three possibilities for their satisfying subintervals: either they are disjoint, or they meet, or they overlap. Each possibility leads to distribution of the invariance requirements over upto three intervals. The D modality can be handled by another set of distribution disjuncts. We do not spell out the rules in detail.

As before, by recursively applying the rules we obtain formulae in bundled *InvHS*. Putting together the rules for all formulae outside the bundled fragment gives an argument that *InvHS* formulae can be equivalently put into its bundled fragment. Hence *InvHS* is expressively complete for $FO^2[<, bet]$. □

5 Drawing some conclusions

As we observed after Definition 1, the logic *InvHS* is semantically no different from *HS*, so Lemma 5 yields an expressiveness result for Halpern-Shoham logic. Venema (1991) showed that *HS* is weaker than full first-order logic, we can now make the weakness precise.

Theorem 6 *HS is expressively complete for* $FO^2[<, bet]$.

We put our expressiveness characterization mentioned in Section 3 to use, by borrowing a result from Krebs et al. (2016). Defining arbitrarily many subintervals within arbitrarily many subintervals is not in general possible in Halpern-Shoham logic:

Corollary 7 *The regular language* $(a(ab)^*b)^+$ *is not definable by an HS formula.*

To further analyze the translation provided in the above result, we measure the size of *HS* formulae using their *dag-size* (see the book of Demri et al. (2016)). That is, we think of a formula as being represented by a directed acyclic graph and count only the number of distinct subformulae of the formula. Each rewrite rule in the translation of Lemma 5 changes dag-size by a constant, so the recursive translation maps an *InvHS* formula to a bundled *InvHS* formula whose dag-size only grows polynomially. Building upon this one obtains a complexity result.

Theorem 8 *The satisfiability problem for HS over finite word models is complete for polynomial space.*

Proof Since $LTL[\mathsf{F}, \mathsf{P}, \mathsf{X}, \mathsf{Y}]$ formulae can be represented in the logic, there is a polynomial space lower bound on the complexity of satisfiability, shown by Sistla and Clarke (1985).

For the upper bound, there is a trivial translation to *InvHS*. The proof of Lemma 5 gives a translation into bundled *InvHS*, which is polynomial in the dag-size of the formula. From the proof of Lemma 4 we can extract a more direct translation from bundled *InvHS*, not into two-variable between logic, but to the point temporal logic of Krebs et al. (2016) (mentioned in the proof of Lemma 2). That paper provides a further polynomial translation to standard linear temporal logic *LTL*. Finally, that the satisfiability of *LTL* formulae can be checked in space polynomial in their dag-size was shown by Sistla and Clarke (1985). □

As we mentioned in the introduction of this paper, in our interval logics, propositions are interpreted at points and not at intervals. Thus we do not encounter the high complexities of two-dimensional HS logics detailed in papers such as Bresolin et al. (2014). The expressiveness of these logics within three-variable first-order logic should be higher, but remains unexplored.

It would be interesting to extend the ideas of this paper to fragments of the Duration Calculus on timed word models. The dissertation of Shah (2012) looks at unary point temporal logics with metric interval modalities.

The work of Wakankar et al. (2017) suggests a different direction, moving from satisfiability problems to those of synthesis. This will require coming up with a suitable notion of automata.

Acknowledgements

Many thanks to Fenrong Liu and Hiroakira Ono for the invitation to speak at the Asian workshop on philosophical logic in Beijing, where the beginnings of this work were presented. Junhua Yu and his team at Tsinghua University did an excellent job with the organization. I would like to thank Yanjing Wang at Peking University for all his help.

Numerous discussions with Paritosh Pandya, our earlier joint work with Simoni Shah, and our recent results with Andreas Krebs and Howard Straubing, have shaped this paper. Several questions by Johan van Benthem at the Beijing AWPL persuaded me that putting together the examples of Section 3 would help illustrate my ideas. Anantha Padmanabha brought to my attention a bundled fragment of his logic, and that led me to expand my logic to full *HS*. I would like to thank the anonymous referee for a careful reading of the paper.

References

Allen, James F. 1983. Maintaining knowledge about temporal intervals. *Commun. Assoc. Comput. Mach.* 26 (11): 832–843.

Andréka, Hajnal, Istvan Németi, and Johan van Benthem. 1998. Modal languages and bounded fragments of predicate logic. *J. Philos. Log.* 27 (3): 217–274.

Basin, David A., and Nils Klarlund. 1995. Hardware verification using monadic second-order logic. In *Proc. 7th CAV, Liège*, ed. Pierre Wolper. Vol. 939 of *LNCS*, 31–41.

Bresolin, Davide, Valentin Goranko, Angelo Montanari, and Guido Sciavicco. 2009a. Propositional interval neighbourhood logics: expressiveness, decidability, and undecidable extensions. *Ann. Pure Appl. Log.* 161 (3): 289–304.

Bresolin, Davide, Dario Della Monica, Valentin Goranko, Angelo Montanari, and Guido Sciavicco. 2009b. Undecidability of interval temporal logics with the overlap modality. In *Proc. 16th TIME, Bressanone-Brixen*, eds. Carsten Lutz and Jean-François Raskin, 88–95.

Bresolin, Davide, Dario Della Monica, Angelo Montanari, Pietro Sala, and Guido Sciavicco. 2014. Interval temporal logics over strongly discrete linear orders: expressiveness and complexity. *Theoret. Comp. Sci.* 560 part 3: 269–291.

Chandra, Ashok, Steven Fortune, and Richard Lipton. 1985. Unbounded fan-in circuits and associative functions. *J. Comp. Syst. Sci.* 30 (2): 222–234.

Demri, Stéphane, Valentin Goranko, and Martin Lange. 2016. *Temporal logics in computer science: finite state systems*. CUP.

Goranko, Valentin, Angelo Montanari, and Guido Sciavicco. 2004. A roadmap of interval temporal logics and duration calculi. *J. Appl. Non-Classical Log.* 14 (1–2): 9–54.

Gurevich, Yuri, and Harry R. Lewis. 1984. A logic for constant-depth circuits. *Inf. Contr.* 61 (1): 65–74.

Halpern, Joseph Y., and Yoav Shoham. 1991. A propositional modal logic of time intervals. *J. Assoc. Comput. Mach.* 38 (4): 935–962.

Halpern, Joseph Y., Zohar Manna, and Ben C. Moszkowski. 1983. A hardware semantics based on temporal intervals. In *Proc. 10th ICALP, Barcelona*, ed. Josep Díaz. Vol. 154 of *LNCS*, 278–291.

Immerman, Neil. 1987. Languages that capture complexity classes. *SIAM J. Comput.* 16 (4): 760–778.

Kamp, Johan Anthony Willem. 1968. Tense logic and the theory of linear order. PhD diss, Dept. Philos., UCLA.

Krebs, Andreas, Kamal Lodaya, Paritosh K. Pandya, and Howard Straubing. 2016. Two-variable logic with a between relation. In *Proc. 31st LICS, New York*, eds. Martin Grohe, Erik Koskinen, and Natarajan Shankar, 106–115. ACM/IEEE.

Krebs, Andreas, Kamal Lodaya, Paritosh K. Pandya, and Howard Straubing. 2018. An algebraic decision procedure for two-variable logic with a between relation. In *Proc. 27th CSL, Birmingham*, eds. Dan Ghica and Achim Jung. Vol. 119 of *LIPIcs*, 28–1. Dagstuhl. 17pp.

Lodaya, Kamal. 2000. Sharperning the undecidability of interval logic. In *Proc. 6th Asian, Penang*, eds. Jifeng He and Masahiko Sato. Vol. 1961 of *LNCS*, 290–298.

Lodaya, Kamal, and Paritosh K. Pandya. 2006. A dose of timed logics. In *Proc. 4th Formats, Paris*, eds. Eugene Asarin and Patricia Bouyer. Vol. 4202 of *LNCS*, 260–273.

Lodaya, Kamal, and Paritosh K. Pandya. 2017. Deterministic temporal logics and interval modalities. In *Proc. 9th Meth. Modalities*, eds. Sujata Ghosh and R. Ramanujam. Vol. 243 of *Eptcs*, 23–40.

Lodaya, Kamal, Paritosh K. Pandya, and Simoni S. Shah. 2008. Marking the chops: an unambiguous temporal logic. In *Proc. 5th IFIP TCS, milano*, eds. Giorgio Ausiello, Juhani Karhumäki, Giancarlo Mauri, and Luke Ong. Vol. 273 of *IFIP*, 461–476.

Lodaya, Kamal, Paritosh K. Pandya, and Simoni S. Shah. 2010. Around dot depth two. In *Proc. 14th DLT, London (CA)*, eds. Yuan Gao, Hanlin Lu, Shinnosuke Seki, and Sheng Yu. Vol. 6224 of *LNCS*, 303–314.

Marcinkowski, Jerzy, and Jakub Michaliszyn. 2011. The ultimate undecidability result for the halpern-shoham logic. In *Proc. 26th LICS, Toronto*, 377–386.

Moszkowski, Ben. 1983. Reasoning about digital circuits. PhD diss, Comp. Sci., Stanford Univ..

Padmanabha, Anantha, R. Ramanujam, and Yanjing Wang. 2018. Bundled fragments of first-order modal logic: (un)decidability. In *Proc. 38th FSTTCS, Ahmedabad*, eds. Sumit Gan-

guly and Paritosh K. Pandya. Vol. 122 of *LIPIcs*, 43–1. Dagstuhl. 20pp.
Pandya, Paritosh K. 2001. Specifying and deciding quantified discrete-time duration calculus formulae using DCVALID: an automata theoretic approach. In *Proc. RTTOOLS, aalborg*.
Pin, Jean-Éric. 1986. *Varieties of formal languages*. Plenum. Translated by A. Howie.
Rabinovich, Alexander. 2000. Expressive completeness of duration calculus. *Inf. Comput.* 156 (1/2): 320–344.
Schwentick, Thomas, Denis Thérien, and Heribert Vollmer. 2002. Partially-ordered two-way automata: a new characterization of DA. In *Proc. 5th DLT, Vienna*, eds. Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa. Vol. 2295 of *LNCS*, 239–250.
Shah, Simoni S. 2012. Unambiguity and timed languages: automata, languages, expressiveness. PhD diss, Tech. Comp. Sci., TIFR.
Sipser, Michael. 1983. Borel sets and circuit complexity. In *Proc. 15th STOC, Boston*, 61–69. ACM.
Sistla, A. Prasad, and Edmund M. Clarke. 1985. The complexity of propositional linear temporal logics. *J. Assoc. Comput. Mach.* 32 (3): 733–749.
Straubing, Howard. 1994. *Finite automata, formal languages, and circuit complexity*. Birkhäuser.
Tesson, Pascal, and Denis Thérien. 2002. Diamonds are forever: the variety DA. In *Semigroups, algorithms, automata and languages*, eds. Gracinda Gomes, Jean-Éric Pin, and Pedro Silva, 475–499. World Scientific.
Tesson, Pascal, and Denis Thérien. 2007. Logic meets algebra: the case of regular languages. *Log. Meth. Comp. Sci.* 3 (1:4): 1–37.
Thérien, Denis, and Thomas Wilke. 1998. Over words, two variables are as powerful as one quantifier alternation. In *Proc. 30th STOC, Dallas*, ed. Jeffrey Vitter, 234–240. ACM.
Thérien, Denis, and Thomas Wilke. 2004. Nesting until and since in temporal logic. *Theory Comput. Syst.* 37 (1): 111–131.
Thomas, Wolfgang. 1997. Languages, automata and logic. In *Handbook of formal language theory III*, eds. Grzegorz Rozenberg and Arto Salomaa. Springer.
van Benthem, Johan. 1983. *The logic of time*. Reidel.
Venema, Yde. 1989. Expressiveness and completeness of an interval tense logic. *Notre Dame J. Formal Log.* 31 (4): 529–547.
Venema, Yde. 1991. A modal logic for chopping intervals. *J. Log. Comput.* 1 (4): 453–476.
Wakankar, Amol, Paritosh K. Pandya, and Raj Mohan Matteplackel. 2017. DCSYNTH: Guided reactive synthesis with soft requirements for robust controller and shield synthesis.
Wilke, Thomas. 1994. Specifying timed state sequences in powerful decidable logics and timed automata. In *Proc. 3rd FTRTFT, Lübeck*, eds. Hans Langmaack, Willem-Paul de Roever, and Jan Vytopil. Vol. 863 of *LNCS*, 694–715.
Zhou, Chaochen, and Michael Hansen. 2004. *Duration calculus*. Springer.
Zhou, Chaochen, Tony Hoare, and Anders Ravn. 1991. A calculus of durations. *Inform. Proc. Lett.* 40 (5): 269–276.