Electronic Notes in Theoretical Computer Science 7 (1997)
URL: http://www.elsevier.nl/locate/entcs/volume7.html ?? pages

On Expressive Completeness of Duration and Mean Value Calculi (Extended Abstract)

Alexander Rabinovich

Computer Science Department
Sackler Faculty of Exact Sciences
Tel Aviv University
Tel Aviv, Israel 69978
e.mail: rabino@math.tau.ac.il

Abstract

This paper compares the expressive power of first-order monadic logic of order, a fundamental formalism in mathematical logic and the theory of computation, with that of two formalisms for the specification of real-time systems, the propositional versions of duration and mean value calculi.

Our results show that the propositional mean value calculus is expressively complete for monadic first-order logic of order. A new semantics for the chop operator used in these real-time formalisms is also proposed, and the expressive completeness results achieved in the paper indicate that the new definition might be more natural than the original one. We provide a characterization of the expressive power of the propositional duration calculus and investigate the connections between the propositional duration calculus and star-free regular expressions. Finally, we show that there exists at least an exponential gap between the succinctness of the propositional duration (mean value) calculus and that of monadic first-order logic of order.

1 Introduction

The Duration Calculus [24] is a formalism for the specification of real time systems. DC is based on interval logic [12,5] and uses real numbers to model time. DC was successfully applied in case studies of software embedded systems, e.g., a gas burner [19], a railway crossing [20] and was used to define the real time semantics of other languages.

A run of a real time system is represented by a function from non-negative reals into a set of values - the instantaneous states of a system. Such a function will be called a signal. Usually, there is a further restriction on the behavior of continuous time systems. For example, a function that assigns value q_0 to the rationals and value q_1 to the irrationals is not accepted as a 'legal' signal.

A requirement that is often imposed in the literature is that in every finite length time interval a system can change its state only finitely many times. This requirement is called non-Zeno (or finite variability) requirement.

Atomic formulas of DC have the form $\lceil S \rceil$, where S is a boolean signal expression. Such a formula has the value true in an interval $[a,\ b]$ if $\int_a^b [\![S]\!]$ is equal to $b-a$, i.e., the signal defined by expression S is true at almost all points of the interval $[a, b]$. If $[\![S]\!]$ denotes a non-Zeno boolean signal $\mathcal{A}$, then this integral condition is equivalent to "$\mathcal{A}$ receives the value false at a finite number of points in the interval $[a,\ b]$."

Note that if $\mathcal{A}$ and $\mathcal{B}$ are non-Zeno signals that disagree only on a finite number of points in any finite interval $[c,\ d]$ (notation $\mathcal{A} \sim_{fin} \mathcal{B}$), then $\int_a^b \mathcal{A} = \int_a^b \mathcal{B}$. The Duration Calculus formulas **respect** $\sim_{fin}$ equivalence, i.e., if $\mathcal{A} \sim_{fin} \mathcal{B}$, then $\mathcal{A}$ satisfies a duration formula D if and only if $\mathcal{B}$ satisfies D. Therefore, in DC it is impossible to specify instantaneous events. In [25], DC was extended to Mean Value Calculus in order to handle instantaneous events.

Expressive completeness is a very important topic in Mathematics, Logics and Computer Science. One of the first theorems that students learn in logic is that negation and conjunction is a complete set of propositional connectives. A classical example for expressive completeness from Computer Science is: a language is accepted by a finite automaton if it is definable by a regular expression.

The examples which are closer to the topics we investigate in this paper are: (1) Kamp's theorem [7] that states that propositional temporal logic has the same expressive power as monadic first order logic over the Dedekind closed linear orders (see also [3,2]). (2) McNaughton's theorem which states that a language is definable by a star free regular expression if and only if it is definable by a monadic formula interpreted over the set of all finite linear orders [10].

In this paper I investigate the expressive power of the Propositional Duration Calculus and of the Propositional Mean Value Calculus. In these fragments the metric aspects of the calculi are ignored. I show

Theorem (Expressive completeness)

(i) First-order monadic logic of order vs PMVC:
  (a) Every PMVC formula is equivalent to a monadic sentence.
  (b) Every monadic sentence is equivalent to a PMVC formula.

(ii) First-order monadic logic of order vs PDC:
  (a) Every PDC formula is equivalent to a monadic sentence which respects $\sim_{fin}$ equivalence.
  (b) Every monadic sentence which respects $\sim_{fin}$ equivalence is equivalent to a PDC formula.

I will show that there exists an exponential gap between the succinctness of monadic logic and that of duration and mean value calculi:

Theorem (Succinctness) There are monadic sentences ψ_n of length $O(log\ n)$ such that ψ_n is not equivalent to any PDC (PMVC) formula of length less than n.

The property specified by ψ_n is very natural; ψ_n is satisfied by a signal if "the signal changes exactly n times".

The duration (metrical) aspects of the Mean Value Calculus and the Duration Calculus are not considered in this paper. These aspects are very important in applications. I tried to understand the logical foundation of these formalisms. The practical applications may require incursions into Calculus (e.g., into differential equations) which have little (if anything) in common with existing well understood tools of Logic and computational model theory. However, the duration free aspects of the Duration Calculus play a very important role in applications. In fact, the majority of the laws and the transformation rules in [18] deal with logical (non-metric) aspects of the duration calculus. Also in [25], nine out of ten axioms for the Mean Value Calculus are duration free (non-metrical).

The rest of the paper is organized as follows. In section 2 definitions are provided and notations and terminology are explained. In section 3 the syntax and the semantics of monadic logic are recalled. In section 4 the syntax and the semantics of duration calculus and of mean-value calculus are provided. Section 5 gives the expressive completeness theorem. Section 6 explains the connection between PDC and star free regular expressions. In section 7 the succinctness results are presented. The definition of the semantics given here for PMVC and PDC differs from that in [6,25]; these differences are not essential for PDC, yet are essential for PMVC. In section 8 the differences are explained and their impact on our main results are discussed. Section 9 states the conclusion and some further results.

The proofs are omitted and will be given in the full version of the paper.

2 Terminology and Notations

$\mathbb{N}$ is the set of natural numbers; $\mathbb{BOOL}$ is the set of booleans and Σ is a finite non-empty set of symbols called an alphabet; we use l, m to range over the elements of Σ. $\mathbb{R}$ is the set of real numbers, $\mathbb{R}^{\geq 0}$ is the set of non-negative reals; a, b will range over $\mathbb{R}^{\geq 0}$; $[a,\ b]$ is a finite length closed interval on the reals; we will use the standard notations for other types of intervals, e.g., $(a,\ b)$ is an open interval; all intervals are assumed to be non-empty sets; I will range over intervals.

A Σ-signal or Σ-predicate over I is a function from I into Σ; the letters $\mathcal{A}$, $\mathcal{B}$ range over Σ-signals. Whenever the domain I and the range Σ of $\mathcal{A}$ is clear from the context we use 'signal' or 'predicate' for 'Σ-predicate over I'. A subset $\mathcal{A}$ of a set I will be identified with the corresponding boolean predicate over I. It is well-known that if Σ is an alphabet of size $n > 1$ and k is the least positive integer such that $n < 2^k$ then Σ-signal can be coded with k boolean predicates.

Definition 2.1 *A function $\mathcal{A}$ from $\mathbb{R}^{\geq 0}$ to Σ is said to be a* non-Zeno *(or* piecewise constant*)* ***Σ-signal*** *if there exists an unbounded increasing sequence $a_0 = 0 < a_1 < a_2 \ldots < a_n < \ldots$ such that $\mathcal{A}$ is constant on every interval (a_i, a_{i+1}).*

In the literature the non-Zeno signals are sometimes called finite variability (or piecewise continuous) trajectories.

A function $\mathcal{B}$ from a subinterval I of $\mathbb{R}^{\geq 0}$ is said to be a non-Zeno Σ-signal over I if $\mathcal{B}$ is the restriction on I of a non-Zeno Σ-signal $\mathcal{A}$. Hence, $\mathcal{B}$ is non-Zeno if it changes its value only a finite number of times in every finite length subinterval of I. In the sequel we will often use the word 'signal' for 'non-Zeno signal'.

A Σ-signal **language** is a set of Σ-signals.

Definition 2.2 *($\sim_{fin}$ equivalence) Signals $\mathcal{A}$ and $\mathcal{B}$ are almost equal (notation $\sim_{fin}$) if for all real numbers a, b the set $\{c \,:\, \mathcal{A}(c) \neq \mathcal{B}(c) \wedge c \in [a,\ b]\}$ is finite. A signal language L* ***respects*** *$\sim_{fin}$ if $\mathcal{A} \sim_{fin} \mathcal{B}$ implies that $\mathcal{A} \in L$ iff $\mathcal{B} \in L$.*

3 Monadic First Order Logic of Order

First-order monadic logic of order is a fundamental formalism in mathematical logic and the theory of computation. We use $X_1, \ldots, X_n$ for monadic predicate symbols and t, u, v for first order variables. We denote by $FOL(X_1, \ldots, X_n,\ <)$ the language of the monadic theory of order with monadic (one-place) predicate symbols $X_1, \ldots, X_n$. The *atomic* formulas of this language are formulas of the form $X_i(t)$ and $t < u$. The formulas are constructed from the atomic formulas by propositional connectives and the first order quantifiers. The notion of free and bound variables is defined as usual; a sentence is a formula without free variables; the quantifier depth of a formula is also defined in the standard way (see e.g. [11]).

A structure for $FOL(X_1,\ X_2, \ldots, X_n,\ <)$ is $K = \langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$, where I is a set partially ordered by $<_I$ and $\mathcal{A}_i$ are subsets of I.

The satisfiability of a formula in a structure $K = \langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$, is defined in the standard way by interpreting X_i by $\mathcal{A}_i$, and $<$ by $<_I$.

In this paper we will consider only substructures of the non-negative reals, i.e., I will be a subinterval of $\mathbb{R}^{\geq 0}$ and $<_I$ will be the restriction on I of the standard order on the reals. We will write $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models \psi$ to indicate that a sentence ψ holds in the structure $\langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$.

We say that a structure $\langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$ is non-Zeno (or piecewise constant) if the characteristic functions of the sets $\mathcal{A}_i$ are non-Zeno signals over I.

4 Duration Calculus and Mean Value Calculus

The propositional duration calculus (PDC) (called the restricted duration calculus in [23,6]) is a fragment of the duration calculus where metric properties are ignored. The propositional mean value calculus [25] is a fragment of the mean value calculus where metric properties are ignored.

In this section the syntax and the semantics of PDC and PMVC are presented. The semantics differ from the semantics given in [6,25] and we will explain the differences and their impact on our results in Section 8.

4.1 Syntax

The sets of formulas of PDC and of PMVC are parameterized by a set $X_1, \ldots, X_n$ of state variables that 'correspond' to the monadic predicates of first order logic.

PDC and PMVC have two syntactical categories: state expressions and formulas.

State Expressions: The *state expressions* are constructed from the state variables by propositional connectives. We will use S to range over the state expressions which are defined by the following grammar:

$$S ::= X \mid S \vee S \mid \neg S, \text{ where } X \text{ is a state variable.}$$

Atomic Formulas of PDC: if S is a state expression, then $\lceil S \rceil$ is an atomic formula of PDC.

Atomic Formulas of PMVC: If S is a state expression, then $\lceil S \rceil^0$ is an atomic formula of PMVC.

The formulas of PMVC and the formulas of PDC are generated in the same way from the atomic formulas.

Formulas: The formulas of PDC (PMVC) are defined by the following grammar:

$D ::= At \mid D ^\frown D \mid \neg D \mid D \vee D$, where At ranges over the atomic formulas of PDC (PMVC, respectively).

4.2 Semantics of PMVC

Let $K = \langle I, \mathcal{A}_1, \ldots, \mathcal{A}_n, < \rangle$ be a non-Zeno structure where I is a subinterval of $\mathbb{R}^{\geq 0}$ and $<$ is the standard order relation on the reals.

The meaning (in K) of state expressions and formulas is provided below.

State Expressions: The meaning $[\![S]\!]^K$ of a state expression S in a structure K is a subset of I defined as usual by the structural induction on the state expressions. Namely,

Variables: $[\![X_i]\!]^K$ is $\mathcal{A}_i$.

Disjunction: $[\![S_1 \vee S_2]\!]^K$ is the union of $[\![S_1]\!]^K$ and $[\![S_2]\!]^K$.

Negation: $[\![\neg S]\!]^K$ is the complement of $[\![S]\!]^K$ relative to I.

It is easy to check that if K is a non-Zeno structure then the characteristic function of $[\![S]\!]^K$ is non-Zeno.

When K is clear from the context we will use $[\![S]\!]$ for $[\![S]\!]^K$; furthermore, we sometimes identify the set $[\![S]\!]$ with its characteristic function.

Below the satisfaction relation between PMVC formulas and the non-Zeno structures is defined.

PMVC Atomic Formulas: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models \lceil S \rceil^0$ iff I is a one point interval $[a, a]$ and $a \in [\![S]\!]$.

The meaning for disjunctions and negation is defined as usual.

$\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D_1 \vee D_2$ iff $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D_1$ or $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D_2$.

$\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models \neg D$ iff not $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D$.

Recall that every interval is a non-empty set and an interval I_1 precedes an interval I_2 if $a \in I_1 \wedge b \in I_2 \rightarrow a < b$. A **chop-partition** of an interval I is an ordered pair of disjoint intervals I_1 and I_2 such that $I = I_1 \cup I_2$ and I_1 precedes I_2. Here are some chop-partitions for the interval $I = [0,\ 1)$:
(1) $I_1 = [0,\ \frac{1}{4}],\ I_2 = (\frac{1}{4},\ 1)$; (2) $I_1 = [0,\ \frac{1}{4}),\ I_2 = [\frac{1}{4},\ 1)$ and (3) $I_1 = \{0\},\ I_2 = (0,\ 1)$. Observe that a one-point interval has no chop-partition.

Chop: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D_1 {}^\frown D_2$ iff $\mathcal{A}_1 \cap I_1, \ldots, \mathcal{A}_n \cap I_1,\ I_1 \models D_1$ and $\mathcal{A}_1 \cap I_2, \ldots, \mathcal{A}_n \cap I_2,\ I_2 \models D_2$ for some chop-partition of I into subintervals $I_1,\ I_2$.

4.3 Semantics of PDC

It will be slightly more convenient for us to define the meaning of PDC only on the structures over positive (including infinite) length intervals.

The meaning of PDC state expressions is defined exactly as the meaning of PMVC state expressions. The satisfaction relation between PDC formulas and the non-Zeno structures over positive length intervals is defined as follows.

PDC Atomic Formulas: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models \lceil S \rceil$ if the complement of $[\![S]\!]$ relative to I does not contain an interval of a positive length.

The meaning for disjunctions and negation is defined as usual. Finally, since PDC is defined only on positive length intervals, in the definition of chops we will take only positive length chop-partitions, namely

Chop: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models D_1 {}^\frown D_2$ if and only if $\mathcal{A}_1 \cap I_1, \ldots, \mathcal{A}_n \cap I_1,\ I_1 \models D_1$ and $\mathcal{A}_1 \cap I_2, \ldots, \mathcal{A}_n \cap I_2,\ I_2 \models D_2$ for some chop-partition of I into positive length subintervals $I_1,\ I_2$.

The following fact is easily proved by the structural induction on PDC formulas.

Fact 4.1 *Let $K = \langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$ and $K' = \langle I,\ \mathcal{A}'_1, \ldots, \mathcal{A}'_n,\ <_I\rangle$ be two structures over the same interval I. Suppose that $\mathcal{A}_i \sim_{fin} \mathcal{A}'_i$ for $i = 1, \ldots, n$. Then $K \models D$ if and only if $K' \models D$ for every PDC formula D.*

Recall that a point a is an internal point of an interval I if there exists $\epsilon > 0$ such that the interval $(a - \epsilon,\ a + \epsilon)$ is a subset of I.

Fact 4.2 *Suppose that I and I' have the same set I^o of internal points. Let $K = \langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$ and $K' = \langle I',\ \mathcal{A}'_1, \ldots, \mathcal{A}'_n,\ <_{I'}\rangle$ be two structures such that $\mathcal{A}_i \cap I^o = \mathcal{A}'_i \cap I^o$ for $i = 1, \ldots n$. Then $K \models D$ if and only if $K' \models D$ for every PDC formula D.*

The $\sim_{fin}$ equivalence on signals (see Definition 2.2) is extended to the structures as follows:

Definition 4.3 *($\sim_{fin}$ equivalence on structures) Let $K = \langle I,\ \mathcal{A}_1, \ldots, \mathcal{A}_n,\ <_I\rangle$ and $K' = \langle I,\ \mathcal{A}'_1, \ldots, \mathcal{A}'_n,\ <_{I'}\rangle$ be two structures. K and K' are $\sim_{fin}$ equivalent (notation $K \sim_{fin} K'$) if*

(i) I and I' have the same set I^o of internal points.

(ii) $\mathcal{A}_i \cap I^o \sim_{fin} \mathcal{A}'_i \cap I^o$ for $i = 1, \ldots, n$.

Definition 4.4 *A formula D respects $\sim_{fin}$ equivalence if from $K \sim_{fin} K'$ it follows that $K \models D$ if and only if $K' \models D$.*

Fact 4.1 and Fact 4.2 imply

Proposition 4.5 *The PDC formulas respect $\sim_{fin}$ equivalence.*

5 Expressive Completeness

Definition 5.1 *A formula ψ_1 is equivalent to a formula ψ_2 over a set of structures CL, if $K \models \psi_1 \iff K \models \psi_2$ for every structure $K \in CL$.*

We say that ψ_1 is equivalent to ψ_2 if ψ_1 is equivalent to ψ_2 over the set of non-Zeno signal structures.

The next theorem is obtained by a direct reformulation of the semantical clauses for PDC and PMVC.

Theorem 5.2 *(Embedding of PDC and PMVC into FOL)*

(i) Every PMVC formula is equivalent to a monadic first-order sentence.

(ii) Every PDC formula is equivalent to a monadic first-order sentence that respects $\sim_{fin}$ equivalence.

Our main theorem is

Theorem 5.3 *(Expressive Completeness)*

(i) For every monadic first-order sentence ψ there is an equivalent PMVC formula D.

(ii) *For every monadic first-order sentence ψ that respects $\sim_{fin}$ there exists a PDC formula D such that ψ and D are equivalent.*

Remark (about the proof of Theorem 5.3.) The proof uses model theoretical techniques developed by Shelah [21] for the decidability of the second order monadic logic (see [4] for a survey of these techniques). The proof will be given in the full version of the paper. Let us only mention that it provides an explicit translation from monadic sentences into PMVC and PDC. A syntactical proof of Theorem 5.3 can be based on the techniques from [16].

Remark It is widely believed that the Mean Value and Duration Calculi cannot specify liveness properties while monadic first-order logic can. This is because the "traditional" semantics of MVC and DC is defined only on the finite length closed subinterval of the reals. The traditional semantics uses the following convention for the interval of all non-negative reals: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ \mathbb{R}^{\geq 0} \models D$ iff $\mathcal{A}_1 \cap I, \ldots, \mathcal{A}_n \cap I,\ I \models D$ for every finite length closed subinterval I of $\mathbb{R}^{\geq 0}$. Theorem 5.3 demonstrates that according to our (more natural) definitions PMVC and PDC are able to express liveness properties.

6 Star Free Expressions and PDC

In this section the relationship between PDC and star free regular expressions is provided. In the first subsection a representation of signals by stuttering free strings is given. In the second subsection the stuttering free interpretation for star free regular expressions is defined and in the third subsection the equivalence between PDC and star free expressions is established.

6.1 Traces of Signals

Definition 6.1 *Σ-signal $\mathcal{A}$ over an interval $[a,\ b)$ is* ***right continuous*** *if there are $a_0 = a < a_1 < \ldots < a_n = b$ such that $\mathcal{A}$ is constant in $[a_i,\ a_{i+1})$ for $i < n$ and $\mathcal{A}(a_i) \neq \mathcal{A}(a_{i+1})$ for $i < n-1$.*

A signal $\mathcal{A}$ over $[a,\ \infty)$ is right continuous if for every $b > a$ the restriction of $\mathcal{A}$ on the interval $[a,\ b)$ is right continuous. The following fact is immediate

Fact 6.2 *For every non-Zeno signal $\mathcal{A}$ over an interval $[a,\ b)$ there exists a unique right continuous signal $\mathcal{B}$ such that $\mathcal{A} \sim_{fin} \mathcal{B}$.*

Definition 6.3 *(Stuttering [8]) A string $l_0 l_1 \ldots l_n$ is stuttering free if $l_i \neq l_{i+1}$ for every $i < n$. A string language is stuttering free if it contains only stuttering free strings. An ω-strings $l_0 l_1 \ldots$ is stuttering free if $\forall i.\ l_i \neq l_{i+1}$.*

Definition 6.4 *(Trace of a signal) Let $\mathcal{A}$ be a right continuous signal over $[a,\ b)$ and let $a_0, \ldots a_n$ be as in Definition 6.1. The trace of $\mathcal{A}$ (notations* ***trace***$(\mathcal{A})$*) is a stuttering free string $l_0 l_1 \ldots l_{n-1}$ such that $l_i = \mathcal{A}(a_i)$, for $i = 0, \ldots, n-1$.*

Traces for non-Zeno signals over the intervals of the form $[a,\ b)$ are defined as follows. Let $\mathcal{A}$ be a non-Zeno signal over $[a,\ b)$. According to Fact 6.2 there exists a unique right continuous signal $\mathcal{B}$ such that $\mathcal{A} \sim_{fin} \mathcal{B}$. The trace of $\mathcal{A}$ is defined as the trace of $\mathcal{B}$.

Let $\langle \mathcal{A}_1, \ldots, \mathcal{A}_n \rangle$ be an n-tuple of boolean signals over an interval I. With this n-tuple we associate a $\{0,1\}^n$ signal $\mathcal{A}$ over I defined as $\mathcal{A}(a) \triangleq \langle \mathcal{A}_1(a) \ldots \mathcal{A}_n(a) \rangle$. The above mapping defines a one-to-one correspondence between the set of n-tuple of boolean signals over I and $\{0,1\}^n$ signal over I. The trace of an n-tuple of boolean signals (notations $\mathbf{trace}(\mathcal{A}_1, \ldots, \mathcal{A}_n)$) is defined as the trace of the corresponding $\{0,1\}^n$ signal.

6.2 Stuttering Free Interpretation of Star Free Expressions

The star free regular expressions over an alphabet Σ are defined by the following grammar: $E ::= \ l \mid E + E \mid E; E \mid \neg E$, where l ranges over Σ. The standard semantics assigns to a star free expression a string language over Σ. In the standard semantics sum $(+)$ is interpreted as union, sequential composition $(;)$ is interpreted as concatenation and negation $(\neg)$ is interpreted as the complementation relative to the set of all finite strings (excluding the empty string ϵ). Recall that McNaughton and Papert proved that a language is definable by a star free regular expression if it is definable by monadic formula interpreted over the set of all finite linear orders [10].

Let us consider stuttering free (see Definition 6.3) interpretations of the negation and of the sequential composition symbols. Namely, let $\neg$ be the complementation relative to the set of stuttering free strings Σ^{stut} and let sequential composition be interpreted as the following operation $\star$ on strings:

$$l_0 \ldots l_p \star m_0 \ldots m_k = \begin{cases} l_0 \ldots l_p m_1 \ldots m_k \ \text{ iff } l_p = m_0 \\ l_0 \ldots l_p m_0 \ldots m_k \text{ otherwise} \end{cases}$$

Sum, like before, is interpreted as union. The stuttering free interpretation assigns to a star free expression E the stuttering free string language which will be denoted by $[\![E]\!]^{stut}$.

6.3 Equivalence between Star Free Expressions and PDC

The concatenation depth of star free expressions is defined as follows: $cd(l) = 0$, $cd(\neg E) = cd(E)$, $cd(E_1 + E_2) = max(cd(E_1),\ cd(E_2))$, and $cd(E_1; E_2) = cd(E_1) + cd(E_2) + 1$. The chops depth of PDC formulas is defined similarly.

Theorem 6.5 *There is an algorithm Tr that maps star free expressions to PDC and there is an algorithm Tr' that maps PDC formulas to star free expressions such that for $a < b \in \mathbb{R}^{\geq 0}$*

(i) The concatenation depth of E is equal to the chop depth of $Tr(E)$.

(ii) The chop depth of D is equal to the concatenation depth of $Tr'(D)$.

(iii) $\mathcal{A}_1, \ldots, \mathcal{A}_n$, $[a,\ b) \models D$ *iff* ***trace***$(\mathcal{A}_1, \ldots, \mathcal{A}_n) \in [\![Tr(D))]\!]^{stut}$.
(iv) $\mathcal{A}_1, \ldots, \mathcal{A}_n$, $[a,\ b) \models Tr'(E)$ *iff* ***trace***$(\mathcal{A}_1, \ldots, \mathcal{A}_n) \in [\![E]\!]^{stut}$.

Remark In the above theorem it is assumed that the alphabet of star free expressions has the size 2^n.

Remark The trace of a right continuous signal over interval $[a,\ \infty)$ is defined similarly to Definition 6.4; it is a finite stuttering free string or a stuttering free ω-string. In the full version of the paper the extension of Theorem 6.5 to infinite length intervals is provided.

7 Succinctness

The next theorem demonstrates that there exists at least an exponential gap between the succinctness of PDC (PMVC) and that of monadic first order logic.

Theorem 7.1 *(Succinctness) There are first order monadic sentences ψ_n, $n \in \mathbb{N}$ such that ψ_n respects $\sim_{fin}$, the length of ψ_n is $O(log\ n)$, however, if a PDC (PMVC) formula D is equivalent to ψ_n then the chop depth of D is at least $n-1$.*

Below we sketch the proof only for the propositional duration calculus. Let C_n be the set of non-Zeno structures $\langle [a,\ b),\ \mathcal{A}, < \rangle$ where $a,\ b \in \mathbb{R}^{\geq 0}$ and $\mathbf{trace}(\mathcal{A})$ is $(01)^n$ (see Definition 6.4). Notice that the property defined by C_n is natural: "A right continuous signal over an interval $[a,\ b)$ is in C_n if it changes its value n times from 0 to 1 in the interval and it is 0 in the beginning of the interval and 1 at its end".

The succinctness theorem is the consequence of the observation that C_n and C_{n+1} are disjoint and of the following two propositions:

Proposition 7.2 *For every n there is a first order monadic formula ψ_n of length $O(log\ n)$ such that $\mathcal{A},\ [a,\ b) \models \psi_n$ if and only if $\langle\ [a,\ b),\ \mathcal{A}, <\ \rangle \in C_n$.*

Proposition 7.3 *Let D be a duration calculus formula of chop depth less than n. If there exists $K \in C_n$ such that $K \models D$, then there exists $K' \in C_{n+1}$ such that $K' \models D$.*

8 Comparison with the Official Semantics

In this section a detailed comparison between our versions of PMVC and PDC semantics and the official semantics (see [25,6]) for PMVC and PDC is provided, and the impact of these differences on our results is discussed.

Below $\models_1$ is used for the satisfaction relations of the PMVC and the PDC versions of [25,6].

First, let us point to the following differences in the semantical definitions:

(i) (*Intervals.*) We provided the semantics of PMVC formulas over arbitrary intervals, whereas in [25] the semantics is given only for the closed intervals of the form $[a,\ b]$. Similarly, we provided the semantics of PDC formulas over all positive length intervals, whereas in [6] the semantics is given only for the intervals of the form $[a,\ b]$ $(a < b \in \mathbb{R}^{\geq 0})$.

(ii) (*Chop.*) Consequently, there is a discrepancy in the definition of chop. Namely, in the official versions $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ [a,\ b] \models_1 D_1 {}^\frown D_2$ if and only if $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ [a,\ m] \models_1 D_1$ and $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ [m,\ b] \models_1 D_2$ for $m \in [a,\ b]$. In contrast to our definition (see Section 4) the intervals $[a,\ m]$ and $[m,\ b]$ are not disjoint, hence $[a,\ m]$, $[m,\ b]$ is not a chop-partition of $[a,\ b]$. Moreover, both these intervals are closed.

It is straightforward to formalize the definitions of the official versions of PMVC and PDC in monadic logic. Therefore, Theorem 5.2 holds for the official versions.

8.1 The Impact on PDC

The difference in the definitions of chop for PDC is not essential. Namely, our satisfaction relation $\models$ agrees with the official satisfaction relation $\models_1$, i.e., $K \models D$ iff $K \models_1 D$, where K is a non-Zeno structure over a closed interval. Hence,

(i) Our definitions conservatively extend the semantics of PDC to arbitrary positive length intervals, and

(ii) The expressive completeness theorem (Theorem 5.3(2)) also holds for the official version of PDC.

8.2 The Impact on PMVC

For PMVC the difference in the definition of chop is essential. Namely,

Proposition 8.1 *There exist a PMVC formula D and signals $\mathcal{A}_1, \ldots, \mathcal{A}_n$ such that*
$\mathcal{A}_1, \ldots, \mathcal{A}_n, [a,\ b] \models D$ holds, however, $\mathcal{A}_1, \ldots, \mathcal{A}_n, [a,\ b] \models_1 D$ fails.

We have not checked yet whether the official version of PMVC is expressively complete in the following sense: for every monadic sentence ψ there exists a PMVC formula D such that $\mathcal{A}_1, \ldots, \mathcal{A}_n, [a,\ b] \models \psi$ if and only if $\mathcal{A}_1, \ldots, \mathcal{A}_n, [a,\ b] \models_1 D$.

Finally, in [25] there are also atomic PMVC formulas of the form [1] $\lceil S \rceil^{\forall}$ with the corresponding semantical clause: $\mathcal{A}_1, \ldots, \mathcal{A}_n,\ I \models_1 \lceil S \rceil^{\forall}$ iff $[\![S]\!]$ is true everywhere in I. An addition of these formulas to PMVC does not increase the

[1] confusingly the notation $\lceil S \rceil$ is used for these formulas in [25].

expressive power because $\lceil\ \rceil^{\forall}$ can be expressed from $\lceil S \rceil^{0}$, boolean connectives and chop [13].

9 Conclusion

In this paper the Propositional Mean Value Calculus was compared to monadic first order logic of order - a very fundamental formalism. Our main result shows that there exist meaning (semantics) preserving translations between PMVC formulas and monadic first order logic. This result confirms that PMVC is not an ad hoc formalism.

Our main result deals with definability in different formalisms. This is completely orthogonal to the decidability issues. In order to conclude that there exists a decision procedure for the equivalence between PMVC(PDC) expressions (or for the satisfiability problem) one can appeal to the decidability of monadic logic over the reals [1] and to (an effective version of) Theorem 5.2. The satisfiability problem for PDC can be also reduced, by Theorem 6.5, to the emptiness problem for star free expressions. However, there is a much simpler way to show decidability of PMVC, PDC and of many other much stronger formalisms (see [14,15]).

Our results hold not only for the reals, however, we still do not have a characterization of linear orders for which PMVC is expressively equivalent to first-order monadic logic (see [16] for generalizations).

It is instructive to compare our completeness result with Kamp's theorem [7,3,2] which states that every monadic formula $\psi(X_1, \ldots X_n, t)$ with one free variable t is equivalent to a propositional temporal logic formula $D(X_1, \ldots X_n)$. Our theorem states that every monadic sentence (formula without free variables) is equivalent to a PMVC formula.

We demonstrated that there exists an exponential gap between the succinctness of PDC (PMVC) and that of monadic logic. The space complexity of the validity problem for all these formalisms has a non-elementary lower bound [22,17]. We believe that the succinctness gap between monadic logic and PDC (PMVC) is much higher than exponential. Probably, the techniques from [9,22] can be used to show that there exists a non-elementary gap in the succinctness.

The definitions of the semantics for PMVC and PDC suggested in this paper differ from those in the official versions [6,25]. The main difference is in the definition of chop. These differences are not essential for PDC because PDC formulas respect $\sim_{fin}$ equivalence. Hence, the expressive completeness theorem holds for the official version of PDC. The difference in the definition of chop is essential for PMVC. We have not checked yet whether the official version of PMVC [25] is expressively complete. We are not aware of any good reason why a more natural semantics for chop suggested here is not used in the MVC official version [25].

Acknowledgement

I would like to thank the anonymous referees for their helpful suggestions.

References

[1] J. Burgess and Y. Gurevich. The decision problem for linear temporal logic. Notre Dame J. Formal Logic, 26(2):115-128, 1985.

[2] D. Gabbay, I. Hodkinson and M. Reynolds. Temporal Logic. Oxford Un. Press, 1994.

[3] D. Gabbay, A. Pnueli, S. Shelah and J. Stavi. On the temporal analysis of fairness. In *7th ACM Symp. on Principles of Prog. Lang.*, pp. 163-173, 1980.

[4] Y. Gurevich. Monadic second order theories. In J. Barwise and S. Feferman eds., Model theoretical logics, pp. 479-506, Springer Verlag.

[5] J. Halperin, B. Moszkowski and Z. Manna. A propositional modal logic of time intervals. In *LICS*, 1986, pp. 279-292.

[6] M. R. Hansen and Zhou Chaochen. Duration Calculus: Logical Foundations. Technical Report, Technical University of Denmark, 1996.

[7] H. Kamp. *Tense logic and the theory of linear order.* Ph.D. Thesis, Un. of California at LA, 1968.

[8] L. Lamport. The Temporal Logic of Actions. ACM Transactions on Programming Languages and Systems, 16(3), pp. 872-923, 1994.

[9] A. R. Meyer and L. Stockmeyer. Non-elementary word problems in automata and logic. In *Proc. AMS Symposium on Complexity of Computation*, 1973.

[10] R. McNaughton and S. Papert. Counter-free automata. The MIT Press, 1971.

[11] J. D. Monk. Mathematical Logic. Springer-Verlag, 1976.

[12] B. Moszkowski. Reasoning about Digital Circuits. Ph. D thesis, Stanford, 1983.

[13] P. Pandya. Some extensions to Propositional Mean Value Calculus: Expressiveness and Decidability. In *Proceedings of Computer Science Logic 1995*, LNCS.

[14] A. Rabinovich. A Universal Continuous Time Specification Formalism. Submitted for publication, Sept. 1996.

[15] A. Rabinovich. On Translation of Temporal Logic of Actions into Monadic Second Order Logic. *To appear in Theoretical Computer Science.*

[16] A. Rabinovich. Star Free expressions over the reals. *To appear in Theoretical Computer Science.*

[17] A. Rabinovich. Non-elementary Lower Bound for Propositional Duration Calculus. Technical Report, Tel-Aviv University, May 1997.

[18] A. Ravn. Design of Embedded Real Time Computing Systems, Technical Report, Technical University of Denmark, 1995.

[19] A. Ravn, H. Richel and K. Hansen. Specifying and verifying requirement of real time systems. *IEEE Transaction on Software Eng.*, 1993.

[20] J. Skakkebak, A. Ravn, H. Richel, Zhou Chaochen. Specification of Embedded Real time Systems. In Proc. of 1992 Euromicro workshop on Real Time Systems. IEEE Computer Society Press.

[21] S. Shelah. The monadic theory of order. *Ann. of Math.*, **102**, pp 349-419, 1975.

[22] L. Stockmeyer. The complexity of decision problems in automata and logic, Ph.D. Thesis, MIT, 1974.

[23] Zhou Chaochen, M. R. Hansen and P. Sestoft. Decidability and undecidablity results for Duration Calculus. In *STACS'93,* Lect. Notes in Comp. Sci. vol 665, pages 58-68, 1993.

[24] Zhou Chaochen, C.A.R. Hoare and A. P. Ravn. A calculus of Duration. *Information processing Letters,* 40(5):269-279, 1991.

[25] Zhou Chaochen and Li Xiaoshan. A mean value calculus of duration. In *A classical Mind: Essays in Honor of C. A. R. Hoare,* pages 431-451, Prentice Hall, 1994.