Fundamenta Informaticae 42 (2000) 1–10
IOS Press

Succinctness Gap between Monadic Logic and Duration Calculus

A. Rabinovich*
Department of Computer Science
Tel Aviv University
Tel Aviv 69978, Israel
e.mail: rabino@math.tau.ac.il

Abstract. In [8, 11] the expressive completeness of the Propositional fragment of Duration Calculus relative to monadic first-order logic of order was established. In this paper we show that there is at least an exponential blow-up in every meaning preserving translation from monadic logic to PDC. Hence, there exists an exponential gap between the succinctness of monadic logic and that of duration calculus.

1. Introduction

The Duration Calculus [2] is a formalism for the specification of real time systems. DC is based on interval logic [4] and uses real numbers to model time. DC was successfully applied in case studies of software embedded systems, e.g., a gas burner [12], a railway crossing [14] 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.

*Address for correspondence: Department of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel

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 finite variability boolean signal η, then this integral condition is equivalent to "η receives the value false at a finite number of points in the interval $[a, b]$."

Note that if η_1 and η_2 are finite variability signals that disagree only on a finite number of points in any finite interval $[c, d]$ (notation $\eta_1 \sim_{fin} \eta_2$), then $\int_a^b \eta_1 = \int_a^b \eta_2$. The Duration Calculus formulas **respect** $\sim_{fin}$ equivalence, i.e., if $\eta_1 \sim_{fin} \eta_2$, then η_1 satisfies a duration formula D if and only if η_2 satisfies D. Therefore, in DC it is impossible to specify instantaneous events. In [11] the expressive completeness of the Propositional fragment of Duration Calculus (PDC) relative to monadic first-order logic of order was established. It was proved there that (1) Every PDC formula is equivalent to a monadic sentence which respects $\sim_{fin}$ equivalence. (2) Every monadic sentence which respects $\sim_{fin}$ equivalence is equivalent to a PDC formula. In [10, 7] it was shown that there exists the meaning preserving translation Tr from PDC to monadic logic of order such that the size of the formula $Tr(D)$ is linear in the size of D. The upper bound for the complexity of our translation algorithm from monadic logic to PDC can be extracted from a careful analysis of the proofs in [11]. The best upper bound which we were able to extract is non-elementary. (Recall that a function f is non-elementary if there is no m such that $f(n)$ is less than $\exp_m(n)$ for all n, where

$$\exp_m(k) \quad = \quad \left. 2^{2^{\cdot^{\cdot^{\cdot^{k}}}}} \right\} m$$

is the m-times iterated exponential function.)

In this paper we show that there is at least an exponential blow-up in every meaning preserving translation from monadic logic to PDC. Hence, there exists an exponential gap between the succinctness of first-order monadic logic and that of duration calculus. The following theorem which was announced in [8] is the main result of this paper.

Theorem There are first order monadic sentences ψ_n, $n > 0$ such that ψ_n respects $\sim_{fin}$, the length of ψ_n is $O(log\ n)$. However, if a PDC formula D is equivalent to ψ_n, then the chop depth of D is at least $n-1$.

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

The space complexity of the validity problems both for first-order monadic logic and for PDC has a non-elementary lower bound [15, 9]. We believe that the succinctness gap between first-order monadic logic and PDC is much higher than exponential.

It is instructive to compare the above completeness and succinctness results with the corresponding results for the Linear Time Temporal Logic. Kamp's theorem [5, 3] states that every first-order monadic formula $\psi(t)$ with one free variable t is equivalent to a propositional temporal logic formula D. (D is equivalent to ψ both over the discrete time model and the real time model.) However, the validity problem for temporal logic has the polynomial space complexity [13]. A closer examination of the proofs in [15, 13] shows that there is a non-elementary gap not only between the complexity of temporal logic and first-order monadic logic, but also in their succinctness.

The paper is organized as follows. In the next section we recall the syntax and semantics of Duration Calculus and state some auxiliary lemmas. Section 3 gives a proof of our main theorem.

2. Propositional Duration Calculus

The Duration Calculus [2] is a formalism for the specification of real time systems. The Propositional Duration Calculus (called the restricted duration calculus in [1]) is a fragment of the duration calculus where metric properties are ignored.

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 gives value q_0 for the rationals and value q_1 for 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 finite variability (or non-Zeno) requirement.

Below we first describe a connection between finite variability functions and stuttering free strings. Then we recall the syntax and the semantics of PDC.

2.1. Finite variability functions

A function η from a subinterval $[a,\ b]$ of the reals into a finite set Σ has finite variability if there exists a finite increasing sequence $a = a_0 < a_1 < a_2 \ldots < a_n = b$ such that η is constant on every interval (a_i, a_{i+1}).

The restriction of η on an interval [c, d] is denoted by $\eta \restriction [c,\ d]$. Notice that if $\eta :\ [a,\ b] \to \Sigma$ has finite variability and $[c,\ d] \subseteq [a,\ b]$, then $\eta \restriction [c,\ d]$ has finite variability.

The following lemma is straightforward.

Lemma 1. *Suppose that $\eta :\ [a,\ b] \to \Sigma$ has finite variability, then there exists a unique increasing sequence $a = a_0 < a_1 < a_2 \ldots < a_n = b$ such that*

1. *η is constant on every interval (a_i, a_{i+1}).*
2. *For every $i < n-1$, the value of η on (a_i, a_{i+1}) differs from the value of η on (a_{i+1}, a_{i+2}).*

Definition 2. (Trace of a finite variability function.) Let η be a finite variability function over $[a,\ b]$ and let $a_0, \ldots, a_n$ be as in Lemma 1. Let l_i be the values of η on (a_i, a_{i+1}). The trace of η (notations $\mathbf{trace}(\eta,\ [a,\ b])$) is the string $l_0 l_1 \ldots l_{n-1}$.

Definition 3.(Stuttering [6]) A string $l_0 l_1 \ldots l_n$ is stuttering free if $l_i \neq l_{i+1}$ for $i < n$. A language is stuttering free if it contains only stuttering free strings.

It is clear that the trace of a signal is a stuttering free string.

The stuttering free concatenation of strings (notation $\star$) is defined as follows:

$$l_0 \dots l_p \star m_0 \dots m_k = \begin{cases} l_0 \dots l_p m_1 \dots m_k & \text{if } l_p = m_0 \\ l_0 \dots l_p m_0 \dots m_k & \text{otherwise.} \end{cases}$$

Lemma 4. *Suppose $\eta : [a,b] \to \Sigma$ and $c \in (a,b)$. Then* **trace** $(\eta, [a,b])$ = **trace** $(\eta, [a,c])$ $\star$**trace** $(\eta, [c,d])$*, where $\star$ is stuttering free concatenation.*

Remark 5. (Trace of a tuple.) Let $\langle \eta_1, \dots, \eta_n \rangle$ be an n-tuple of finite variability functions from $[a,\ b]$ into $\{0,\ 1\}$. With this n-tuple we associate a function η from $[a,\ b]$ into $\{0, 1, \dots, 2^n - 1\}$ defined as $\eta(t) = i$ if $\langle \eta_1(t), \dots, \eta_n(t) \rangle$ is the binary representation of i. The above mapping defines a one-to-one correspondence between the set of n-tuple of finite variability functions from $[a,\ b]$ into $\{0, 1\}$ and finite variability functions from $[a,\ b]$ into $\{0, \dots, 2^n - 1\}$. The trace of an n-tuple (notations $\mathbf{trace}(\eta_1, \dots, \eta_n)$) is defined as the trace of the corresponding function.

2.2. Syntax of PDC

PDC 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:

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

Atomic Formulas of PDC: if S is a state expression, then $\lceil S \rceil$ is an atomic formula of PDC.
Formulas: The formulas of PDC are defined by the following grammar:

$$D ::= At \mid D \frown D \mid \neg D \mid D \vee D \mid D \wedge D, \text{ where } At \text{ ranges over the atomic formulas of PDC.}$$

The binary operation $\frown$ is called chop. The chop rank of PDC formulas is defined as follows: $cr(At) = 0$ for atomic formulas; $cr(\neg D) = cn(D)$, $cr(D_1 \vee D_2) = cr(D_1 \wedge D_2) = max(cr(D_1),\ cr(D_2))$ and $cr(D_1 \frown D_2) = cr(D_1) + cr(D_2) + 1$.

2.3. Semantics of PDC

A valuation θ over an interval $[a,\ b]$ is a function that assigns to every state variable X a finite variability function from $[a,\ b]$ into $\{0,\ 1\}$. A valuation θ straightforwardly extends to state expressions using the meaning of the propositional connectives pointwise. We use the notation $[\![S]\!]^{DC}\theta$ for the function assigned to the state expression S under the valuation θ. It is clear that $[\![S]\!]^{DC}\theta$ has finite variability.

The satisfaction relation $\models$ between PDC formulas and valuations over a positive length interval $[a,\ b]$ is defined as follows:

PDC Atomic Formulas: $\theta,\ [a,\ b] \models \lceil S \rceil$ if there is no positive length subinterval of $[a,\ b]$ where $[\![S]\!]^{DC}\theta$ is constant and equal to 0.

Propositional Connectives: The meaning for disjunction, conjunction and negation is defined as usual.
$\theta, [a,\ b] \models D_1 \vee D_2$ iff $\theta, [a,\ b] \models D_1$ or $\theta, [a,\ b] \models D_2$.
$\theta, [a,\ b] \models D_1 \wedge D_2$ iff $\theta, [a,\ b] \models D_1$ and $\theta, [a,\ b] \models D_2$.
$\theta, [a,\ b] \models \neg D$ iff not $\theta, [a,\ b] \models D$.

Let us denote by $\theta \restriction [c,\ d]$ the valuation that maps every state variable X to the restriction of $\theta(X)$ on $[c,\ d]$.

Chop: $\theta,\ [a,\ b] \models D_1 \frown D_2$ if $\theta \restriction [a,\ m],\ [a,\ m] \models D_1$ and $\theta \restriction [m,\ b],\ [m,\ b] \models D_2$ for some $m \in (a,\ b)$.

Definition 6.($\sim_{fin}$ equivalence.) Signals η_1 and η_2 over the same subinterval are said to be $\sim_{fin}$-equivalent if every finite length interval has only finitely many points where η_1 is not equal to η_2. Valuations θ_1 and θ_2 are said to be $\sim_{fin}$ equivalent if for every state variable X, the signals $\theta_1(X)$and $\theta_2(X)$ are $\sim_{fin}$ equivalent.

The following lemmas are proved by the structural induction on PDC formulas.

Lemma 7. *If θ_1 is $\sim_{fin}$ equivalent to θ_2, then $\theta_1,\ [a,\ b] \models D$ if and only if $\theta_2,\ [a,\ b] \models D$.*

Lemma 8. *If* **trace**$(\theta_1,\ [a_1,\ b_1])$=**trace**$(\theta_2,\ [a_2,\ b_2])$, *then $\theta_1,\ [a_1,\ b_1] \models D$ if and only if $\theta_2,\ [a_2,\ b_2] \models D$.*

PDC formulas D^1 and D^2 are said to be equivalent if $\theta, [a,\ b] \models D^1$ whenever $\theta, [a,\ b] \models D^2$.

The following lemma follows from the definition of chop, Lemma 4 and Lemma 8.

Lemma 9. *1. If $\theta_1, [a_1, b_1] \models D_1$ and $\theta_2, [a_2, b_2] \models D_2$ and* **trace** $(\theta, [a, b])$ = **trace**$(\theta_1, [a_1, b_1])\star$ **trace** $(\theta_2, [a_2, b_2])$, *then $\theta, [a, b] \models D_1 \frown D_2$.*

2. *If $\theta, [a, b] \models D_1 \frown D_2$ then there is $m \in (a, b)$ such that $\theta, [a, m] \models D_1$ and $\theta, [m,\ b] \models D_2$ and* **trace**$(\theta, [a,\ b])$ =**trace**$(\theta, [a,\ m])\star$ **trace**$(\theta, [m,\ b])$.

3. Succinctness Gap

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

Theorem 10. (Succinctness) There are first order monadic sentences ψ_n, $n > 0$ such that ψ_n respects $\sim_{fin}$, the length of ψ_n is $O(log\ n)$, however, if a PDC formula D is equivalent to ψ_n, then the chop depth of D is at least $n - 1$.

Let D_n be the PDC formula

$$\underbrace{\lceil \neg X \rceil ^\frown \lceil X \rceil ^\frown \ldots ^\frown \lceil \neg X \rceil ^\frown \lceil X \rceil}_{n \text{ times}}.$$

The concatenation rank of D_n is $2n-1$. It is easy to check that $\theta,\ [a,\ b] \models D_n$ if and only if $\mathbf{trace}(\theta,\ [a,\ b]) = (01)^n$. We will show that every PDC formula equivalent to D_n has the chop rank at least $n-1$, while there is a monadic first order formula of length $O(logn)$ which is equivalent to D_n. Therefore, there exists at least an exponential gap between the succinctness of monadic logic and that of the duration calculus.

Notice that the property defined by D_n is natural.

Theorem 10 immediately follows from Proposition 11 and Proposition 12.

Proposition 11. For every n there is a first order monadic formula ψ_n of length $O(log\ n)$ which is equivalent to D_n.

Proof:
Let

$$Const_1(X, t_1,\ t_2) \triangleq t_1 < t_2 \wedge \forall t.\ t_1 < t < t_2 \rightarrow X(t)$$

$$Const_0(X, t_1,\ t_2) \triangleq t_1 < t_2 \wedge \forall t.\ t_1 < t < t_2 \rightarrow \neg X(t)$$

$$Jump_{0\rightarrow 1}(X, t) \triangleq \exists t_1 t_2.\ t_1 < t < t_2 \wedge Const_0(X, t_1, t) \wedge Const_1(t, t_2)$$

$$Jump_{1\rightarrow 0}(X, t) \triangleq \exists t_1 t_2.\ t_1 < t < t_2 \wedge Const_1(X, t_1, t) \wedge Const_0(t, t_2)$$

$$Jump(X, t) \triangleq Jump_{0\rightarrow 1}(X, t) \vee Jump_{1\rightarrow 0}(X, t)$$

Below we define formulas $\chi_n(X, t_1,\ t_2)$ which say that "there are n jumps of X from 0 to 1 in an interval $[t_1,\ t_2)$ and the first and the last jumps of X are from 0 to 1". The definition of χ_{2n} will contain one occurrence of χ_n and a constant number of other symbols. This will insure that the length of χ_n grows like $log(n)$. Such a trick was used by Albert R. Meyer and Larry Stockmeyer (see e.g., [15]).

$$\begin{aligned}\chi_1(X, t_1,\ t_2) \triangleq \exists t.\ \ & t_1 < t < t_2 \wedge Jump_{0\rightarrow 1}(X, t) \wedge \\ & \forall t'.\ ((t_1 < t' < t_2 \wedge Jump(X, t')) \rightarrow t = t').\end{aligned}$$

For $n > 0$ define $\chi_{2n}(X, t_1,\ t_2)$ as

$$\begin{aligned}&\exists t.\ \ t_1 < t < t_2 \wedge Jump_{1\rightarrow 0}(X, t) \wedge \\ &\quad \forall t_3 t_4.[(t_3 = t_1 \wedge t_4 = t) \vee (t_3 = t \wedge t_4 = t_2)] \rightarrow \chi_n(X, t_3,\ t_4).\end{aligned}$$

For $n > 0$ define $\chi_{2n+1}(X, t_1,\ t_2)$ as

$$\exists t.\ t_1 < t < t_2 \wedge Jump_{1\to 0}(X,t) \wedge \chi_{2n}(X, t_1,\ t) \wedge \chi_1(X, t, t_2).$$

It is easy to see that the length of χ_n is $O(log\ n)$. Finally, define $\psi_n(X)$ as

$$\exists t_1 t_2.\ \chi_n(X,\ t_1,\ t_2) \wedge \forall t.\ (t \leq t_1 \vee t \geq t_2) \rightarrow \neg Jump(X,t).$$

It is easy to verify that ψ_n satisfies Proposition 11. $\square$

Proposition 12. If a duration calculus formula D is equivalent to D_n, then its chop rank is at least n.

Proof. Recall that the trace of a finite variability signal is a stuttering free string. A stuttering free string over $\{0,\ 1\}$ has one of the four following forms: $(01)^n,\ (10)^n, 1(01)^n,\ 0(10)^n$.

Let C_n^i (for $i = 0, \ldots, 3$ and $n > 0$) be the stuttering free string defined as follows:

$$C_n^i = \begin{cases} (01)^n & \text{if } i = 0 \\ (10)^n & \text{if } i = 1 \\ 0(10)^n & \text{if } i = 2 \\ 1(01)^n & \text{if } i = 3 \end{cases}$$

Recall that $\theta,\ [a,\ b] \models D_n$ if and only if $\mathbf{trace}(\theta,\ [a,\ b]) = C_n^0$. Proposition 12 immediately follows from the following

Lemma 13. Let D be a duration calculus formula of chop rank at most $n-1$ and let k be greater than or equal to n. Assume that $\mathbf{trace}(\theta_1,\ [a,\ b]) = C_k^i$ and $\mathbf{trace}(\theta_2,\ [c,\ d]) = C_{k+1}^i$. Then, $\theta_1,\ [a,\ b] \models D$ if and only if $\theta_2,\ [c,\ d] \models D$.

Proof:
(of Lemma 13) We say that a formula D distinguishes between strings s_1 and s_2 if there are $\theta_1,\ [a_1,\ b_1]$ and $\theta_2,\ [a_2,\ b_2]$ such that $\mathbf{trace}(\theta_1,\ [a_1,\ b_1]) = s_1$ and $\mathbf{trace}(\theta_2,\ [a_2,\ b_2]) = s_2$ and $\theta_1,\ [a_1,\ b_1] \models D$, while not $\theta_2,\ [a_2,\ b_2] \models D$. We say that a set Φ of formulas distinguishes between s_1 and s_2 if there is a formula $D \in \Phi$ that distinguishes between s_1 and s_2. It is clear that if Φ cannot distinguish between s_1 and s_2, then the set of boolean combinations of formulas from Φ cannot distinguish between s_1 and s_2.

Below we will show by induction on n that the set of formulas of chop rank $< n-1$ cannot distinguish between C_k^i and C_{k+1}^i for $k \geq n$. This implies Lemma 13.

First, note that without the loss of generality we may assume that X is the only state variable that occurs in these formulas.

Induction Basis. Observe the every atomic formula with the state variable X is equivalent to one of the following four formulas: $\lceil \neg X \rceil$, $\lceil X \rceil$ $\lceil X \vee \neg X \rceil$, $\lceil X \wedge \neg X \rceil$. These formulas cannot distinguish between C_k^i and C_{k+1}^i for $k \geq 1$. Hence their boolean combinations cannot distinguish between C^i_k and C^i_{k+1}. Since every formula of the chop rank 0 is equivalent to a boolean combination of the atomic formulas this proves the inductive basis.

Inductive Step. Assume that the Lemma was proved for n. Let us show that no formula D of the form $D_1 \frown D_2$ whose chop rank is at most n cannot distinguish between C^i_k and C^i_{k+1} for $k > n$. This will imply that no formula of the chop rank at most n can distinguish between C^i_k and C^i_{k+1} for $k > n$.

Assume that $\theta,\ [a,\ b] \models D_1 \frown D_2$ and $\mathbf{trace}(\theta,\ [a,\ b]) = C^i_r$, where $r \in \{k,\ k+1\}$. Then, there is $m \in (a,\ b)$ such that $\theta,\ [a,\ m] \models D_1$ and $\theta,\ [m,\ b] \models D_2$.

We examine separately cases when $i = 0, \dots, 3$. Each of these cases has a number of subcases. The verification of all these subcases is simple but tedious and relies on Lemma 9 which will be used without references.

Case $i = 0$. In this case $C^i_k = (01)^k$ and $C^i_{k+1} = (01)^{k+1}$. Therefore, one of the following subcases holds:

Subcase $\mathbf{trace}(\theta,\ [a,\ m]) = (01)^{r_1}$ *and* $\mathbf{trace}(\theta,\ [m,\ b]) = (01)^{r-r_1}$ *and* $r_1, r - r_1 > 0$.
If $r = k$ (respectively, $r = k+1$) then either (1) the chop rank of D_1 is less than r_1 (respectively $r_1 - 1$) and hence by inductive hypothesis D_1 cannot distinguish between $(01)^{r_1}$ and $(01)^{r_1+1}$ (respectively between $(01)^{r_1-1}$ and $(01)^{r_1}$), or (2) the chop rank of D_2 is less than $r - r_1$ (respectively, $r - r_1 - 1$) and hence by the inductive hypothesis D_2 cannot distinguish between $(01)^{r_2}$ and $(01)^{k_2 - r_1}$ (respectively, between $(01)^{r-r_1}$ and $(01)^{r-r_1} - 1$). In both cases by the definition of chop and Lemmas 9 we obtain that $D_1 \frown D_2$ cannot distinguish between $(01)^k$ and $(01)^{k+1}$.

Subcase $\mathbf{trace}(\theta,\ [a,\ m]) = 0$ *and* $\mathbf{trace}(\theta,\ [m,\ b]) = (01)^r$.
This subcase follows from the inductive hypothesis because D_2 cannot distinguish between $(01)^r$ and $(01)^{r\pm 1}$.

Subcase $\mathbf{trace}(\theta,\ [a,\ m]) = 0$ *and* $\mathbf{trace}(\theta,\ [m,\ b]) = 1(01)^{r-1}$.
This subcase follows from the inductive hypothesis because D_2 cannot distinguish between $(01)^{r-1}$ and $(01)^{r-1\pm 1}$.

Subcase $\mathbf{trace}(\theta,\ [a,\ m]) = (01)^r$ *and* $\mathbf{trace}(\theta,\ [m,\ b]) = 1$.
This subcase follows from the inductive hypothesis because D_1 cannot distinguish between $(01)^r$ and $(01)^{r\pm 1}$.

Subcase $\mathbf{trace}(\theta,\ [a,\ m]) = 0(10)^{r-1}$ *and* $\mathbf{trace}(\theta,\ [m,\ b]) = 1$.
This subcase follows from the inductive hypothesis because D_1 cannot distinguish between $0(10)^{r-1}$ and $0(10)^{r-1\pm 1}$.

Subcase $\mathbf{trace}(\theta, [a, m]) = 0(10)^{r_1}$ *and* $\mathbf{trace}(\theta, [m, b]) = (01)^{r-r_1}$ *and* $r_1, r - r_1 > 0$.
This subcase is similar to the first subcase.

Subcase $\mathbf{trace}(\theta, [a, m]) = (01)^{r_1}$ *and* $\mathbf{trace}(\theta, [m, b]) = 1(01)^{r-r_1}$ *and* $r_1, r - r_1 > 0$.
This subcase is similar to the first subcase.

Subcase **trace**$(\theta, [a, m]) = 0(10)^{r_1}$ *and* **trace**$(\theta, [m, b]) = 1(01)^{r-1-r_1}$ *and* $r_1, r-1-r_1 > 0$.

This subcase is similar to the first subcase.

Case $i = 1$. In this case $C^i_k = (10)^k$ and $C^i_{k+1} = (10)^{k+1}$. This case is dual to the case $i = 0$.
Case $i = 2$. In this case $C^i_k = 0(10)^k$ and $C^i_{k+1} = 0(10)^{k+1}$. Here we consider eight subcases.

Subcase **trace**$(\theta,\ [a,\ m]) = 0$ *and* **trace**$(\theta,\ [m,\ b]) = (10)^r$.

This subcase follows from the case $i = 0$.

Subcase **trace**$(\theta,\ [a,\ m]) = 0$ *and* **trace**$(\theta,\ [m,\ b]) = 0(10)^r$.

This subcase follows from the inductive hypothesis.

Subcase **trace**$(\theta,\ [a,\ m]) = (01)^r$ *and* **trace**$(\theta,\ [m,\ b]) = 0$.

This subcase follows from the case $i = 0$.

Subcase **trace**$(\theta,\ [a,\ m]) = 0(10)^r$ *and* **trace**$(\theta,\ [m,\ b]) = 0$.

This subcase follows from the inductive hypothesis.

Subcase **trace**$(\theta,\ [a,\ m]) = 0(10)^{r_1}$ *and* **trace**$(\theta,\ [m,\ b]) = (10)^{r-r_1}$ *and* $r, r - r_1 > 0$.

This subcase is similar to the first subcase of the case $i = 0$.

Subcase **trace**$(\theta,\ [a,\ m]) = 0(10)^{r_1}$ *and* **trace**$(\theta,\ [m,\ b]) = 0(01)^{r-r_1}$ *and* $r, r - r_1 > 0$.

This subcase is similar to the first subcase of the case $i = 0$.

Subcase **trace**$(\theta,\ [a,\ m]) = (01)^{r_1}$ *and* **trace**$(\theta,\ [m,\ b]) = 0(10)^{r-r_1}$ *and* $r, r - r_1 > 0$.

This subcase follows from the inductive hypothesis.

Subcase **trace**$(\theta,\ [a,\ m]) = (01)^{r_1}$ *and* **trace**$(\theta,\ [m,\ b]) = 1(01)^{r-r_1}$ *and* $r, r - r_1 > 0$.

This subcase follows from the inductive hypothesis.

Case $i = 3$. This case is dual to the case $i = 2$.

This completes the proof of Lemma 13 and of Proposition 12. □

Remark 14. Proposition 12 can be slightly improved. Namely, it can be shown that no formula of the chop rank $< 2n - 1$ is equivalent to D_n. In order to prove this, instead of Lemma 13, one can prove that no formula of the chop rank $< 2n - 1$ can distinguish between C^i_k and C^i_{k+1} for $i = 0, 1$ and $k > n$ and no formula of the chop rank $< 2n$ can distinguish between C^i_k and C^i_{k+1} for $i = 2, 3$ and $k > n$.

References

[1] Zhou Chaochen and M. Hansen. Duration Calculus: Logical Foundations. In *Formal Aspects of Computing*, 9:283-330, 1997.

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

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

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

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

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

[7] P. Pandya. Some extensions to Propositional Mean Value Calculus: Expressiveness and Decidability. In *Proceedings of Computer Science Logic 1995*, LNCS vol. 1092, 1995.

[8] A. Rabinovich. On expressive completeness of Duration and Mean Value Calculi. In *Proceedings of EXPRESS*, Electronic Notes in Theoretical Computer Science, Vol. 7, 1997.

[9] A. Rabinovich. Non-elementary Lower Bound for Propositional Duration Calculus. *Information Processing Letters* 66 (1998), 7-11.

[10] A. Rabinovich. On the Decidability of Continuous Time Specification Formalisms. *Journal of Logic and Computation*, Vol. 8, No 5, pp 669-678, 1998.

[11] A. Rabinovich. Expressive Completeness of Duration Calculus. *Information and Computation,* Vol. 156, No. 1/2, pp. 320-344, 2000.

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

[13] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. J. ACM 32(3):733-749, 1985

[14] 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.

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