A Sampling Semantics of Duration Calculus

Dang Van Hung and Phan Hong Giang *

International Institute for Software Technology
The United Nations University, P.O.Box 3058, Macau

Abstract. In the paper, the problem of discretization of continuous time in Duration Calculus (DC) is addressed. For a DC formula D, for a sampling step h, a sampling semantics $[[D]]_h$ is defined to express the properties of discrete observations of its models, while original semantics $[[D]]$ expresses the properties of the models. In practice, only sampling semantics is implemented. So, an implementation D of a system specified by S, where both are written in DC, is correct iff $[[D]]_h \subseteq [[S]]$. Some rules for proving the correctness of an implementation are given. The problem of digitization is also considered in the paper. Some forms of digitizable DC formulas are shown. Then we apply the obtained results to a non-trivial example, namely, a Biphase Mark Protocol introduced in [11]. That protocol uses 18-cycle cell for one bit of message. A cell is formed by 5-cycle mark subcell followed by 13-cycle code subcell. We adopt the same assumptions about physical environment as in [11]. However, we are able to show a stronger property than in [11]: The protocol works correctly provided the ratio of the writer's and reader's clock is within 30%.

1 Introduction

In real-time hybrid systems, we have to reason about their timed behavior. Since in such systems we have to deal with analog components whose states change continuously, and with digital components (computer programs) whose states change only at ticks of the computer clock, we have to deal with both continuous time and discrete time, and also with the computer clock rate. It would be interesting if these things could be handled by the same formal tool when reasoning about real-time hybrid system. In this paper, we consider how they are being handled in the framework of Duration Calculus (DC) [4].

Consider, for example, a design of a hybrid control system which consists of a control program and a physical plant interacting at discrete sampling time points (see, e.g. [13, 14]). The design aims to derive a specification of the control program from the specification of the continuous behavior of the plant and the environment. The design is correct if, under the assumption of the behavior of the environment, the derived 'discrete' specification implies the specification of the continuous behavior of the plant. In other words, it should be the case that

* On leave from the Institute of Information Technology, Nghia Do, Tu Liem, Hanoi, Vietnam. Email: dvh@iist.unu.edu

the continuous behavior of the plant will satisfy the requirement if the discrete design is satisfied for the discrete model of time formed by all the sampling time points. Since in DC we can adopt both a continuous semantics model and a sampling one [5], we can use a DC formula for continuous time to specify the behavior of the plant and the environment and a DC formula for discrete time to specify the control program. Therefore, we have to define how a DC formula in discrete time implies a DC formula in continuous time. To do so, in this paper, besides the satisfaction for a model of a DC formula in continuous time, we define the satisfaction for the same model of a DC formula in discrete time with respect to a sampling step. In this way we define a sampling semantics for DC formulas and its relation to the original semantics. Hence, we can reason about the correctness of design and implementation of a hybrid system in a uniform manner. The design process of a hybrid system in this framework can be formulated as follows. For a specification S, at the first step we refine it into a formula D such that $D \Rightarrow S$ in DC, then we find another DC formula E and a sampling step h such that the sampling semantics of E w.r.t. h "implies" the usual semantics of D. If necessary, E is refined further in "discrete" DC; this refinement therefore corresponds to the development of a computer program.

An interesting problem in discretization of continuous time is the digitizability of specification formulas. A principle for this has been introduced in [10], that in some cases reduces the verification problem with a dense time domain to a verification problem with a discrete time domain. We consider the digitizabilty of DC formulas and unfortunately find that the formulas involving the operator *chop* (;) and $\int$, in general, are not digitizable; only the formulas expressed in the metric of time are digitizable.

The paper is organized as follows. In the next section, a brief summary of DC is presented. Section 3 introduces sampling semantic model of DC. In Section 4, we consider the power of sampling semantics when the sampling step approaches 0. The digitizability of DC formulas is discussed in Section 5. The last section is devoted to an example: using DC and its continuous and sampling semantics for the specification and verification of a Biphase Mark Protocol.

2 Duration Calculus

In this section, we give a brief summary of DC and its application to specification of real-time systems. For more details, readers are referred to [4].

Time in DC is the set $\mathbb{R}^+$ of nonnegative real numbers. For $t, t' \in R^+, t \leq t', [t, t']$ denotes the time interval from t to t'. Let $intv$ denote the set of all time intervals.

Syntax of DC

Assume that $V = \{X, Y, \ldots\}$ is a set of state variables. We will use the special symbol ℓ to denote the length of an interval, f_i^n, A_i^n $(i, n \geq 0)$ as n-ary function and n-ary predicate names respectively.

The set of state expressions is generated by

$$P \mathrel{\widehat{=}} 0 \mid 1 \mid X \mid \neg P \mid P \vee P \ ,$$

where X stands for a state variable.

The set of terms is generated by

$$r \mathrel{\widehat{=}} \textstyle\int P \mid \ell \mid f(r_1, \ldots, r_n) \ ,$$

where P stands for a state expression, n for a natural number, and f for an n-ary function name.

If A_i^n is an n-ary predicate name, and $r_1, \ldots, r_n$ are n terms then $A_i^n(r_1, \ldots, r_n)$ is an atomic formula.

The set of formulas is generated by the grammar:

$$D \mathrel{\widehat{=}} A \mid D; D \mid \neg D \mid D \vee D \ ,$$

where A stands for an atomic formula of DC.

Semantics of DC

Assume that each n-ary function name f_i^n is associated with a total function from $\mathbb{R}^n$ to $\mathbb{R}$ which is denoted by f_i^n also, and each n-ary predicate name A_i^n is associated with a total function from $\mathbb{R}^n$ to $\{ tt, ff\}$ which is also denoted by A_i^n. In this paper, for simplicity we interpret the functions f as operators on real, e.g. $+, *$, and the relations A as comparison operators between reals, e.g. $<, \leq, =, >, \geq$.

An interpretation $\mathcal{I}$ is a function $\mathcal{I} \in (V \to (\mathbb{R}^+ \to \{0,1\}))$, for which each $\mathcal{I}(X)$, $X \in V$ has at most finitely many discontinuity points in any interval $[a, b]$. We shall use the abbreviation $X_{\mathcal{I}} \mathrel{\widehat{=}} \mathcal{I}(X)$.

The semantics of a state expression P in an interpretation $\mathcal{I}$ is a function $\mathcal{I}_P \in Time \to \{0,1\}$ defined inductively on the structure of state expressions by:

$$\begin{array}{lll}
\mathcal{I}_0(t) \mathrel{\widehat{=}} 0\,, & & \mathcal{I}_{(\neg P)}(t) \mathrel{\widehat{=}} 1 - \mathcal{I}_P(t)\,, \\
\mathcal{I}_X(t) \mathrel{\widehat{=}} X_{\mathcal{I}}(t)\,, & & \\
\mathcal{I}_1(t) \mathrel{\widehat{=}} 1\,, & & \mathcal{I}_{(P \vee Q)}(t) \mathrel{\widehat{=}} \begin{cases} 0 \text{ if } \mathcal{I}_P(t) = 0 \text{ and } \mathcal{I}_Q(t) = 0\,, \\ 1 \text{ otherwise.} \end{cases}
\end{array}$$

The semantics of a term r in an interpretation $\mathcal{I}$ is a function $\mathcal{I}_r \in intv \to \mathbb{R}$ defined inductively on the structure of terms by:

$$\begin{array}{rlrl}
\mathcal{I}_{\int P}([a,b]) & \mathrel{\widehat{=}} \int_a^b \mathcal{I}_P(t)dt\,, & \mathcal{I}_\ell([a,b]) & \mathrel{\widehat{=}} b - a\,, \\
\mathcal{I}_{f_i^n(r_1,\ldots,r_n)}([a,b]) & \mathrel{\widehat{=}} f_i^n(\mathcal{I}_{r_1}([a,b]), \ldots, \mathcal{I}_{r_n}([a,b])) \ . & &
\end{array}$$

The semantics $[[D]]$ of a formula D is a function from the set of interpretation $\mathcal{I}$ and the set $intv$ to $\{ tt, ff\}$, defined inductively on the structure of formulas as follows. Using the abbreviations $\mathcal{I}, [a,b] \models D \mathrel{\widehat{=}} [[D]]([a,b]) = tt$ and $\mathcal{I}, [a,b] \not\models D \mathrel{\widehat{=}} [[D]]([a,b]) = ff$, $[[D]]$ is defined by:

$$\begin{array}{ll}
\mathcal{I}, [a,b] \models A_i^n(r_1, \ldots, r_n) \text{ iff } A_i^n(\mathcal{I}_{r_1}([a,b]), \ldots, \mathcal{I}_{r_n}([a,b])) = tt\,, & \\
\mathcal{I}, [a,b] \models true\,, & \\
\mathcal{I}, [a,b] \models (\neg D) & \text{iff } \mathcal{I}, [a,b] \not\models D\,, \\
\mathcal{I}, [a,b] \models (D_1 \vee D_2) & \text{iff } \mathcal{I}, [a,b] \models D_1 \text{ or } \mathcal{I}, [a,b] \models D_2\,, \\
\mathcal{I}, [a,b] \models (D_1; D_2) & \text{iff } \mathcal{I}, [a,m] \models D_1\,, \text{ and} \\
 & \quad \mathcal{I}, [m,b] \models D_2 \text{ for some } m \in [a,b] \ .
\end{array}$$

For a DC formulas D, D_1 and a state expression P, we use the following abbreviations:

$$\begin{array}{rcl@{\qquad\qquad}rcl} \Diamond D & \widehat{=} & (true; D); true\,, & \Box D & \widehat{=} & \neg(\Diamond(\neg D))\,, \\ \lceil P \rceil & \widehat{=} & \int P = \ell \wedge \ell > 0\,, & \lceil\ \rceil & \widehat{=} & \ell = 0\,, \\ D \Rightarrow D_1 & \widehat{=} & (\neg D) \vee D_1\,, & D \wedge D_1 & \widehat{=} & \neg((\neg D) \vee (\neg D_1))\ . \end{array}$$

Some axioms and theorems in the calculus are as follows.

1. (**DA 1**) $\int 0 = 0$.
2. (**DA 2**) For an arbitrary state P, $\int P \geq 0$.
3. (**DA 3**) For arbitrary states P and Q, $\int P + \int Q = \int(P \vee Q) + \int(P \wedge Q)$.
4. (**DA 4**) Let P be a state and r, s nonnegative real numbers.
   $(\int P = r + s) \Leftrightarrow (\int P = r; \int P = s)$.
5. (Monotonicity) If $D_1 \Rightarrow A_1$ and $D_2 \Rightarrow A_2$ then $D_1; D_2 \Rightarrow A_1; A_2$.
6. (Associativity) $(D_1; D_2); D_3 \Leftrightarrow D_1; (D_2; D_3)$.
7. $P \Rightarrow Q$ then $\lceil P \rceil \Rightarrow \lceil Q \rceil$.
8. $(\lceil P \rceil \wedge \lceil Q \rceil) \Rightarrow \lceil P \wedge Q \rceil$.
9. (Induction rules) Let X be a formula letter occurring in the formula $D(X)$ and let P be a state. Then
   If $D(\lceil\ \rceil)$ holds and $D(X \vee X; \lceil P \rceil) \wedge D(X \vee X; \lceil \neg P \rceil)$ are provable from $D(X)$, then $D(\ true)$ holds,
   If $D(\lceil\ \rceil)$ holds and $D(X \vee \lceil P \rceil; X) \wedge D(X \vee \lceil \neg P \rceil; X)$ are provable from $D(X)$, then $D(\ true)$ holds.

Example 1. To conclude this section, we give an example of using DC in specifying a real-time system. The example is a simple gas burner taken from [4]. One of the time-critical requirements of a gas burner is specified by a DC formula denoted by **Req-1**, defined as

$$\textbf{Req-1} \qquad\qquad \ell > 60s \Rightarrow (20 * \int leak \leq \ell)\ .$$

This says that if the interval over which the system is observed is at least one minute, the proportion of time spent in the leak state is not more than one-twentieth of the elapsed time. The requirement is refined into two design decisions

$$\begin{array}{ll} \textbf{Des-1} & \Box(\lceil leak \rceil \Rightarrow \ell \leq 1s)\ , \\ \textbf{Des-2} & \Box(\lceil leak \rceil; \lceil \neg leak \rceil; \lceil leak \rceil \Rightarrow \ell \geq 30s)\ . \end{array}$$

Des-1 says that any leak state must be detected and stopped within one second, and **Des-2** says that leak must be separated by at least 30s. The design is shown to be correct by proving the implication

$$\textbf{Des-1} \wedge \textbf{Des-2} \Rightarrow \textbf{Req-1}\ .$$

3 Sampling Semantics

We refer to the semantics of DC formulas given in the previous section as continuous semantics or analog semantics. The continuous semantics is good for the specification and the design of the system at abstract levels, since the real systems are physical devices whose states are changing continuously. However, computer programs work in a discrete time manner. Thus, in the refinement of specifications, we need a "discrete semantics" model of DC formulas as well. The sampling semantics of DC formulas is defined in this section. For a DC formula D, for a positive real number h, the sampling semantics of D, denoted by $[[D]]_h$, is a mapping from the set of interpretations of state variables and the set of intervals *intv* to { *ff,tt*}. We will use the abbreviation:

$$\mathcal{I}, [a, b] \models_h D \ \widehat{=} \ [[D]]_h(\mathcal{I}, [a, b]) = tt \ .$$

Let $\mathcal{I}$ be an interpretation over state variables. The time model of sampling with sampling step h is the set $R_h \ \widehat{=} \ \{ih \mid i = 0, 1, 2, \ldots\}$. In this paper from now on, we assume that every state variable is interpreted as a right-side continuous boolean function in the interpretation $\mathcal{I}$.

The semantics of a term r in the interpretation $\mathcal{I}$ is a function $\mathcal{I}_r^h$ from *intv* to reals, defined inductively on the structure of terms in the same way as $\mathcal{I}_r$, except that $\mathcal{I}^h_{\int P}$ and $\mathcal{I}^h_\ell$ are defined as follows.

$$\mathcal{I}^h_{\int P}([a, b]) \widehat{=} h \sum_{j=i}^{k-1} \mathcal{I}_P(jh) \,,$$
$$\mathcal{I}^h_\ell([a, b]) \widehat{=} (k - i)h \,,$$

where $i = [a/h]_{0.5}$, $k = [b/h]_{0.5}$, and for any real number x, ϵ ,

$$[x]_\epsilon \widehat{=} \text{ if } x \leq \lfloor x \rfloor + \epsilon \text{ then } \lfloor x \rfloor \text{ else } \lfloor x \rfloor + 1 \ .$$

Intuitively, $\mathcal{I}^h_{\int P}$ approximates the integral of the state variable P using an algebraic sum, ih and kh are approximations of the time points a and b, respectively, in the discrete model of time.

The semantics of formulas is defined exactly in the same way as in the previous section.

It follows immediately from the definition that not every axiom and rule of DC is sound in the sampling semantic model. For the axioms and rules mentioned in the previous section, the axioms **DA 4**, the induction rule 9 are not sound, the others are sound. The axiom **DA4** now is replaced by the following:

$$\int P = mh + nh \ \Leftrightarrow \ \int P = mh; \int P = nh \,,$$

where $m, n \in \mathbb{N}$ of natural numbers.

Since the term ℓ/h ranges over the set $\mathbb{N}$, we can use the natural induction rule on natural numbers instead of the induction rule based on the finite variability of states. The natural induction rule is formulated as:

Let D be a DC formula. Then if D holds for $\ell = 0$, and if $(\ell = (k+1)h) \Rightarrow D$ is provable from $(\ell = kh) \Rightarrow D$ then $\Box D$ holds.

In general $[[D]] \neq [[D]]_h$. In the sequel, we consider the relationship between them.

Definition 1. *For a state variable P and a positive real number δ, we say P is δ- stable if δ-stable(P) is satisfied, where*

$$\delta\text{-}stable(P) \;\widehat{=}\; \Box(\lceil\neg P\rceil; \lceil P\rceil; \lceil\neg P\rceil \Rightarrow \ell \geq \delta) \;.$$

The formula δ-stable(P) says that the state P keeps *true* for at least δ time units whenever it becomes *true*.

Theorem 1. *Let $\delta > h$. Then*

1. $\dfrac{\mathcal{I}, [a,b] \models \ell = d}{\mathcal{I}, [a,b] \models_h |\ell - d| < h}$, $\qquad \dfrac{\mathcal{I}, [a,b] \models_h \ell = d}{\mathcal{I}, [a,b] \models |\ell - d| < h}$,
2. $\dfrac{\mathcal{I}, [a,b] \models \delta\text{-}stable(P) \wedge \int P = d}{\mathcal{I}, [a,b] \models_h |\int P - d| \leq \min(\ell, (\frac{\ell}{\delta} + 1)h)}$,
   $\dfrac{\mathcal{I}, [ih,kh] \models \delta\text{-}stable(P) \wedge \int P = d}{\mathcal{I}, [ih,kh] \models_h |\int P - d| \leq \min(\ell, \frac{\ell}{\delta}h)}$.

Intuitively, item 1 says that the time metric is approximated with a tolerance less than the length of the sampling step, while duration of a stable state variable is approximated with the tolerance depending on the length of observation time interval, degree of its stability and the length of the sampling step.

Proof. Suppose that $t_0, \ldots, t_n$ are all the discontinuity points of $\mathcal{I}_P$ in the interval $[a,b]$ (note: the state variables in DC are finitely variable). Since δ-stable(P) is true for interval $[a,b]$, it follows that $t_m - t_{m-2} \geq \delta$, $m = 2, \ldots, n$. Hence, $n \leq 2 \times \frac{b-a}{\delta}$. Furthermore, for $j = i, \ldots, k-1$,

$$|\int_{jh}^{(j+2)h} P dt - hP(jh)| \;\begin{cases} = 0 \text{ if } \forall m.t_m \notin [jh, (j+2)h]\,, \\ \leq h \text{ otherwise} \;. \end{cases}$$

Since $|a - ih| + |b - kh| < h$, we have $|\int_a^b P(t)dt - h\sum_{j=i}^{k-1} P(jh)| \leq h(\frac{b-a}{\delta} + 1)$. □

Corollary 1. *Let i, k be in $\mathbb{N}$, $i \leq k$, a, b, h, δ be in $\mathbb{R}^+$, $a < b$, $\delta > h$. Then*

1. $\dfrac{\mathcal{I}, [a,b] \models \delta\text{- }stable(P),\ \mathcal{I}, [a,b] \models_h \int P = d}{\mathcal{I}, [a,b] \models |\int P - d| \leq h(\frac{\ell}{\delta} + 1)}$
2. $\dfrac{\mathcal{I}, [a,b] \models \delta\text{- }stable(P),\ \mathcal{I}, [a,b] \models_h \int P < d}{\mathcal{I}, [a,b] \models \int P < d + h(\frac{\ell}{\delta} + 1)}$
3. $\dfrac{\mathcal{I}, [a,b] \models \delta\text{- }stable(P),\ \mathcal{I}, [a,b] \models \int P < d}{\mathcal{I}, [a,b] \models_h \int P < d + h(\frac{\ell}{\delta} + 1)}$

4. $$\frac{\mathcal{I},[a,b]\models \delta\text{- }stable(P),\ \mathcal{I},[a,b]\models_h \int P>d}{\mathcal{I},[a,b]\models \int P>d-h(\frac{\ell}{\delta}+1)}$$

5. $$\frac{\mathcal{I},[a,b]\models \delta\text{- }stable(P),\ \mathcal{I},[a,b]\models \int P>d}{\mathcal{I},[a,b]\models_h \int P>d-h(\frac{\ell}{\delta}+1)}$$

6. $$\frac{\mathcal{I},[ih,kh]\models \delta\text{- }stable(\neg P),\ \mathcal{I},[ih,kh]\models_h \lceil P\rceil}{\mathcal{I},[ih,kh]\models \lceil P\rceil}$$

7. $$\frac{\mathcal{I},[a,b]\models \delta\text{- }stable(\neg P),\ \mathcal{I},[a,b]\models_h \lceil P\rceil}{\mathcal{I},[a,b]\models \Diamond(\lceil P\rceil\wedge \ell\geq b-a-h)}$$

8. $$\frac{\mathcal{I},[ih,kh]\models \lceil P\rceil}{\mathcal{I},[ih,kh]\models_h \lceil P\rceil}$$

9. $$\frac{\mathcal{I},[a,b]\models \delta\text{- }stable(\neg P)\wedge\delta\text{- }stable(P),\ \mathcal{I},[a,b]\models_h \lceil P\rceil;\lceil\neg P\rceil}{\mathcal{I},[a,b]\models \Diamond((\lceil P\rceil;\lceil\neg P\rceil)\wedge \ell\geq b-a-h)}$$

10. $$\frac{\begin{array}{c}\mathcal{I},[ih,kh]\models \delta\text{- }stable(\neg P)\wedge\delta\text{- }stable(P)\\ \mathcal{I},[ih,kh]\models \lceil P\rceil\wedge\ell\geq h;\lceil\neg P\rceil\wedge\ell\geq h\end{array}}{\mathcal{I},[ih,kh]\models_h (\lceil P\rceil;\lceil\neg P\rceil)}\ .$$

Definition 2. *For DC formulas D, D', we say*

$[[D]]\subseteq[[D']]_h$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, if $\mathcal{I},[a,b]\models D$ then $\mathcal{I},[a,b]\models_h D'$,*
$[[D]]_h\subseteq[[D']]$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, if $\mathcal{I},[a,b]\models_h D$ then $\mathcal{I},[a,b]\models D'$,*
$[[D]]_h\equiv[[D']]$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, $\mathcal{I},[a,b]\models_h D$ if and only if $\mathcal{I},[a,b]\models D'$.*

Example 2. It follows immediately from the definition of sampling semantics that:

1. $[[\ell=d]]\subseteq[[|\ell-d|\leq h]]_h$, $[[\ell=d]]_h\subseteq[[|\ell-d|\leq h]]$,
2. For all state expression Q, we have $[[\lceil Q\rceil]]\subseteq[[|\int Q-\ell|\leq h]]_h$.

Theorem 2. *For any DC formulas D_1, D_2 and D_1', D_2',*

1. $$\frac{[[D_1]]_h\subseteq[[D_1']],\ [[D_2]]_h\subseteq[[D_2']]}{\begin{array}{c}[[D_1;D_2]]_h\subseteq[[D_1';D_2']],\ \ [[D_1\vee D_2]]_h\subseteq[[D_1'\vee D_2']],\\ [[D_1\wedge D_2]]_h\subseteq[[D_1'\wedge D_2']]\end{array}},$$

2. $$\frac{[[D_1]]\subseteq[[D_1']]_h,\ [[D_2]]\subseteq[[D_2']]_h}{\begin{array}{c}[[D_1;D_2]]\subseteq[[D_1';D_2']]_h,\ [[D_1\vee D_2]]\subseteq[[D_1'\vee D_2']]_h,\\ [[D_1\wedge D_2]]\subseteq[[D_1'\wedge D_2']]_h\end{array}}.$$

Note that for a DC formula D, if $[[D]]_h\subseteq[[D]]$, then any model that satisfies D in the sampling model of time satisfies D in continuous time as well. Inversely, if $[[D]]\subseteq[[D]]_h$, then any model that satisfies D in continuous time satisfies D in the sampling model of time as well. Following [10], for the latter, we say that D is closed under sampling w.r.t. h, and for the former, we say that D is closed under inverse sampling w.r.t. h.

Theorems presented in this section make the assumption that state expressions involving integral need to be stable. In practice, this assumption is often satisfied by environment from the physical laws, e.g. the water level cannot change too fast, the wind should change its direction and speed slowly, etc.

Suppose that we want to verify that a system (an interpretation $\mathcal{I}$) satisfies a specification (a DC formula D) under an assumption (a DC formula A), that is to verify $\mathcal{I} \models A \Rightarrow D$. One method of doing so is to find formulas ψ, ϕ such that, $[[A]] \subseteq [[\psi]]_h$, $[[\phi]]_h \subseteq [[D]]$ and prove that $\mathcal{I} \models_h \psi \Rightarrow \phi$. The problem of verification in the integer model of time is much easier, and in many cases, the satisfiability of discrete DC formulas is decidable (see [6]). The theorems presented above give some guide for finding such formulas.

Example 3. Consider the simple gas burner in Example 1. From Example 2 we have that $[[\lceil\ leak\rceil]] \subseteq [[|\int\ leak - \ell| \leq h]]_h$, and $[[\ell \leq 1 - h]]_h \subseteq [[\ell \leq 1]]$. Hence, if we can prove that

$$\mathcal{S} \models_h \Box(|\int\ leak - \ell| \leq h \Rightarrow \ell \leq 1 - h)\,, \tag{1}$$

then we can conclude

$$\mathcal{S} \models \Box(\lceil\ leak\rceil \Rightarrow \ell \leq 1)\ .$$

Thus, if $h \leq 1/2$, and we can ensure that if *leak* at jh then *non-leak* at $(j+1)h$ for all natural j (this means $\mathcal{S} \models_h \lceil\ leak\rceil \Rightarrow \ell = h$) then (1) holds. Note that this can be proved by using natural induction rule of discrete DC.

4 Limit of Sampling Semantics

Since for any fixed interval $[a, b]$, $\lim_{h\to 0} \mathcal{I}^h_{\int P}([a, b]) = \mathcal{I}_{\int P}([a, b])$, we expect that the class of the formulas D, such that for an interval $[a, b]$, there is a real number δ for which $h < \delta \Rightarrow [[D]]_h \subseteq [[D]]$, is large enough to cover many formulas encountered in practice. This means that for sampling steps that are small enough, the sampling semantics and continuous semantics are approximative, and we can replace one by the other. In fact, when specifying a system, we use the continuous semantics, while in implementing the system, we use the sampling semantics. To ensure that the implementation is correct, it must be the case that the sampling semantics of the implemented formula implies the continuous semantics of the specification formula of the system. A question arises as whether there exists such an implementation. Or, more precisely, given a satisfiable DC formula S, do there exist a sampling step h and a satisfiable formula D such that $[[D]]_h \subseteq [[S]]$?

In this section, we define another kind of semantics of DC formulas that we call limit semantics that is useful for the answer to this question.

For a DC formula D, the limit semantics of D, denoted by $[[D]]_l$, is a mapping from the set of interpretations of state variables and the set *intv* of intervals to $\{$ *tt,ff*$\}$, is defined by:

$[[D]]_l(\mathcal{I},[a,b]) = tt$ iff there exists a sequence h_n such that $\lim_{n\to\infty} h_n = 0$ and for all n, $[[D]]_{h_n}(\mathcal{I},[a,b]) = tt$. In other words, $\mathcal{I},[a,b] \models_l D$ if there exists a sequence h_n such that $\lim_{n\to\infty} h_n = 0$ and for all n, $\mathcal{I},[a,b] \models_{h_n} D$.

As in the previous section, we use the denotation:

Definition 3. *For DC formulas D, D', we say*

$[[D]] \subseteq [[D']]_l$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, if $\mathcal{I},[a,b] \models D$ then $\mathcal{I},[a,b] \models_l D'$,*
$[[D]]_l \subseteq [[D']]$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, if $\mathcal{I},[a,b] \models_l D$ then $\mathcal{I},[a,b] \models D'$,*
$[[D]]_l \equiv [[D']]$ *iff for all interpretations $\mathcal{I}$, for all intervals $[a,b]$, $\mathcal{I},[a,b] \models_l D$ if and only if $\mathcal{I},[a,b] \models D'$.*

From the properties of limits, the following theorem is obvious.

Theorem 3.

$$[[\textstyle\int P \le d]]_l \subseteq [[\int P \le d]]\,,\ [[\int P < d]] \subseteq [[\int P < d]]_l\,,$$
$$[[\textstyle\int P < d]]_l \subseteq [[\int P \le d]]\,,\ [[\int P \ge d]]_l \subseteq [[\int P \ge d]]\,,$$
$$[[\textstyle\int P > d]] \subseteq [[\int P > d]]_l\,,\ [[\int P > d]]_l \subseteq [[\int P \ge d]]\ .$$

Now we want to compare different kinds of semantics for more general formulas. The theorem guides us to consider those formulas in which all inequalities evolving integrals of state variables are either of the form $t_1 > t_2$ or of the form $t_1 \le t_2$. In terms of topology, the formulas are either open or closed.

Formally, we define open formulas and closed formulas as follows.

Definition 4. *Continuous terms are defined recursively by:*

- *$\int P$ is a continuous term, where P is a state variable,*
- *real constants are continuous terms,*
- *if $t_1,\ldots,t_n$ are continuous terms, and if f_n is always interpreted as a n-ary continuous function, then $f_n(t_1,\ldots,t_n)$ is a continuous term.*

Closed (open) formulas are defined by:

- *if t_1, t_2 are continuous terms, then $t_1 \le t_2$ ($t_1 < t_2$) is a closed (open) formula,*
- *if D_1, D_2 are closed (open) formulas, then $D_1 \vee D_2$, $D_1 \wedge D_2$, $D_1; D_2$ are closed (open) formulas.*

Example 4. The formula *true* is a closed formula, since *true* $\widehat{=}$ $\int 1 \ge 0$, and the formula $\lceil L \rceil \Rightarrow \ell < 1$ is an open formula, since

$$\lceil L \rceil \Rightarrow \ell < 1 \ \equiv\ (\textstyle\int 1 < 1 \vee \int 1 > \int L)\,.$$

Theorem 4. *For a closed formula D, $[[D]]_l \subseteq [[D]]$, and for an open formula D, $[[D]] \subseteq [[D]]_l$.*

The meaning of the theorem is that given a closed formula D, if for sampling steps which are small enough, D is satisfied by a model by sampling, then we can conclude that D is satisfied by the model; inversely, for an open formula D and a model of it, we can find a sampling step h such that D is satisfied by the model by sampling. The proof of the theorem is immediate from the following lemmas. In fact, we prove a strengthening of the theorem.

Lemma 1. *For an interpretation $\mathcal{I}$, for an interval $[a, b]$, for a closed formula D and sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ such that $h_n \to 0,\ a_n \to a,\ b_n \to b$, if for all n $\mathcal{I}, [a_n, b_n] \models_{h_n} D$ then $\mathcal{I}, [a, b] \models D$.*

Proof. The proof is by induction on the structure of the closed formulas. Note that for any continuous term t, for any interval $[a, b]$ and sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ such that $h_n \to 0,\ a_n \to a,\ b_n \to b$, it holds that $\lim_{n\to\infty} \mathcal{I}_t^{h_n}([a_n, b_n]) = \mathcal{I}_t([a, b])$.

1. Let $D \mathrel{\widehat{=}} t_1 \leq t_2$, where t_1 and t_2 are continuous terms. It follows immediately from the properties of limit that for any interval $[a, b]$ and sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ as in the lemma, if for all n $\mathcal{I}_{t_1}^{h_n}([a_n, b_n]) \leq \mathcal{I}_{t_2}^{h_n}([a_n, b_n])$, then $\mathcal{I}_{t_1}([a, b]) \leq \mathcal{I}_{t_2}([a, b])$.
2. Let $D \mathrel{\widehat{=}} D_1 \vee D_2$, where D_1, D_2 are closed formulas. Let $[a, b]$ be an interval, sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ be as above. Assume that for all n, $\mathcal{I}, [a_n, b_n] \models_{h_n} D_1 \vee D_2$. Then, there are $i \in \{1, 2\}$ and subsequences h_{n_k}, a_{n_k} and b_{n_k} such that $\mathcal{I}, [a_{n_k}, b_{n_k}] \models_{h_{n_k}} D_i$. By the inductive hypothesis, $[[D_i]]_l \subseteq [[D_i]]$, which implies $[[D_1 \vee D_2]]_l \subseteq [[D_1 \vee D_2]]$.
3. Let $D \mathrel{\widehat{=}} D_1; D_2$, where D_1, D_2 are closed formulas. Let $[a, b]$ be an interval, sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ be as above. Assume that for all n $\mathcal{I}, [a_n, b_n] \models_{h_n} D_1; D_2$. Then, there is a sequence $\{c_n\}$ such that for all n $a_n \leq c_n \leq b_n$, $\mathcal{I}, [a_n, c_n] \models_{h_n} D_1$ and $\mathcal{I}, [c_n, b_n] \models_{h_n} D_2$. From the boundedness of $\{c_n\}$, there are a number $c \in [a, b]$ and a subsequence $\{c_{n_k}\}$ such that $c_{n_k} \to c$. By the inductive hypothesis, $\mathcal{I}, [a, c] \models D_1$ and $\mathcal{I}, [c, b] \models D_2$.
4. The case $D \mathrel{\widehat{=}} D_1 \wedge D_2$ is proved in similar way.

□

Lemma 2. *For any interpretation $\mathcal{I}$, for any interval $[a, b]$, for any open formula D if $\mathcal{I}, [a, b] \models D$, then for all sequences $\{h_n\}, \{a_n\}$ and $\{b_n\}$ such that $h_n \to 0,\ a_n \to a,\ b_n \to b$, there is a natural number N such that $\mathcal{I}, [a_n, b_n] \models_{h_n} D$ for all $n \geq N$.*

Proof. Taking into account that for any interpretation $\mathcal{I}$, for any interval $[a, b]$, for any continuous term t $\lim_{h\to 0} \mathcal{I}_t^h([a, b]) = \mathcal{I}_t([a, b])$, the proof is similar to the previous one and is omitted here. □

It follows from the theorem that if a specification of a system is an open formula then with a sampling step that is small enough, if the specification is satisfied by the system in the real time model then it also is satisfied by the system in the sampling time model. Thus, an open specification is not satisfiable if it is not satisfiable in every sampling model of time. For a closed specification, if it

is satisfied by a system in every sampling time model then it also is satisfied by the system in the real-time model.

According to Theorem 9, in order to reduce the problem of verifying $\mathcal{I} \models A \Rightarrow D$ to the problem of verifying $\mathcal{I} \models_h A \Rightarrow D$, the formula A should be an open formula, and D should be a closed formula. Otherwise, we have to weaken the assumption A into an open formula and strengthen the conclusion D into a closed formula.

5 Digitization

Now assume that there is a digital clock that ticks at all time points jh, $j = 0, 1, \ldots$. Consider the sampling model of time according to the digital clock. Then, the time model is the set of non negative integers $\mathbb{N} = \{0, 1, 2, \ldots\}$. It can be seen from the previous sections that the sampling semantics can be useful only under the condition that the state variables are δ-stable for some δ that is large enough relative to the time unit. Otherwise, it is inadequate to describe the behavior of real-time systems. For example, the very simple formula $\lceil P \rceil$ can be true in the sampling model of time for a model in which it fails in the continuous model of time.

Thus, a richer discrete semantics model is needed to represent the digitization. In digitization, we consider the state changes between two consecutive digital clock ticks as they were at the latter tick, and a model is approximated by its digitization in this way. The sampling model w.r.t. the sampling step 1 of a state essentially is the digitization of the state with the digital clock starting at time 0. The digitizability is to reduce the problem of verification in the real-time model to the one in the integer time model. In [10], a digitization quantum is introduced to define the digitizability. According to their definition, a property (formula) D is digitizable iff for all models $\mathcal{I}$, $\mathcal{I}$ satisfies D if and only if for any digitization quantum $0 \leq \epsilon < 1$, the digitization $\mathcal{I}$ by a digital clock starting from time ϵ (with the sampling step $h = 1$) satisfies D. In other words, D is digitizable iff it can be observed by any digital clock running at the rate 1. This kind of observation approximates the metric between two discontinuity points of any state variable by a value with a tolerance less than 1. Thus, many properties specified by Metric Temporal Logic formulas are digitizable. However, for a property specified with duration variables, this kind of observation cannot, in general, approximate duration variables by the values with a tolerance $\delta < 1$ as we can see in many examples. In [2] the authors tried to generalize the results for the duration variables. However, the results obtained on digitization are very weak and work for a very restrict class of properties only. Thus, we cannot expect results as presented in [10] for Duration Calculus.

However, since in Duration Calculus, we can express the properties of the metric of time, we follow the definition of digitizability from [10], and expect the properties about the order of events, length of intervals, and stability are digitizable.

For every interpretation $\mathcal{I}$ of state variables, for every $0 \leq \epsilon < 1$, an integer interpretation $\mathcal{I}^\epsilon$ derived from $\mathcal{I}$ by observing $\mathcal{I}$ with the ϵ-digital clock is defined as follows. For each state variable P, since $\mathcal{I}_P$ is finitely variable, the set of its discontinuity points is finite or countable. Let $t_0, t_1, \ldots$ be the sorted sequence of the discontinuity points of $\mathcal{I}_P$. Then, $[t_0]_\epsilon, [t_1]_\epsilon, \ldots$ is sorted as well. Then, for any $i \in \mathbb{N}$, $[t_i]_\epsilon \leq t < [t_{i+1}]_\epsilon$,

$$\mathcal{I}^\epsilon_P(t) = 1 \text{ iff } \mathcal{I}_P(t') = 1 \text{ for some } t' \text{ such that } t_i < t' < t_{i+1} \ .$$

From the definition, for any state variable P, $\mathcal{I}^\epsilon_P$ is a boolean function which is continuous from the right and can be discontinuous only at integer points.

Example 5. Let $\mathcal{I}_P$ be expressed by the lower part in Fig. 1, where the discontinuity points of $\mathcal{I}_P$ are $0.7, 1.8, 2.65, 4.3$. Let $\epsilon = 0.6$. Then $\mathcal{I}^\epsilon_P$ is expressed by the upper part.

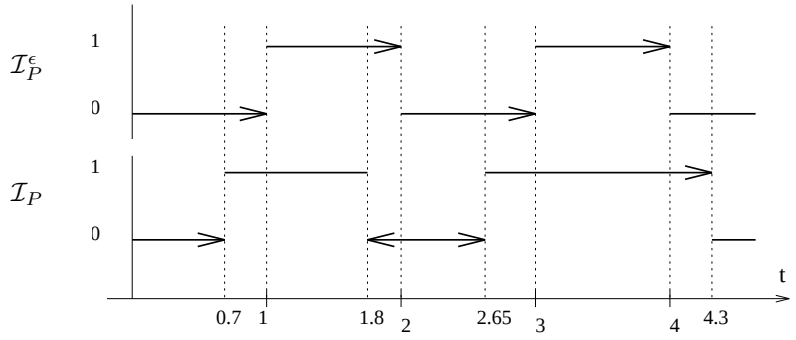

Fig. 1. Digitization of a state

Definition 5. *A DC formula D is said to be digitizable iff for any interpretation $\mathcal{I}$, for any interval $[a, b]$,*

$$\mathcal{I}, [a, b] \models D \text{ if and only if } \mathcal{I}^\epsilon, [[a]_\epsilon, [b]_\epsilon] \models_1 D \text{ for all } 0 \leq \epsilon < 1 \ .$$

Example 6.

1. The formula $D \mathrel{\widehat{=}} 1 < \int P \leq 2$ is not digitizable. Let $\mathcal{I}$ be an interpretation in which P is interpreted as the boolean function

$$\mathcal{I}_P(t) \mathrel{\widehat{=}} \begin{cases} 1 \text{ if } \exists n \in \mathbb{N}. n \leq t \leq n + 0.5 \\ 0 \text{ otherwise} . \end{cases}$$

   Then, for all $n \in \mathbb{N}$, $I, [n, n+4] \models D$. However, for any $0 \leq \epsilon < 1$, either $I^\epsilon, [n, n+4] \models_1 \int P = 4$ or $I^\epsilon, [n, n+4] \models_1 \int P = 0$.

2. The formula $D \mathrel{\widehat{=}} \ell \leq 5$ is digitizable, since for any $a, b, a \leq b$, $b - a \leq 5$ if and only if for any $0 \leq \epsilon < 1$, $b_\epsilon - a_\epsilon \leq 5$.

In general, we have the following:

Theorem 5.

1. *For any state variable P, $\lceil P \rceil^* \mathrel{\widehat{=}} \lceil P \rceil \vee \lceil \ \rceil$ is digitizable.*
2. *For any integer k, $\ell \leq k$, $k \leq \ell$ are digitizable.*
3. *If A, B are digitizable then $A \wedge B$ is digitizable.*

Proof. First we prove item 1. Recall that we have assumed that interpretations of state variables are boolean functions that are continuous from the right. Thus, for any $a \leq b$, $\lceil P \rceil^*$ is *true* if and only if the function $\mathcal{I}_P(t)$ has no discontinuity point in (a, b). The function $\mathcal{I}_P(t)$ has no discontinuity point in (a, b) if and only if for any $0 \leq \epsilon < 1$, by the definition of $\mathcal{I}^\epsilon$, the boolean function $\mathcal{I}^\epsilon_P(t)$ has no discontinuity point in $([a]_\epsilon, [b]_\epsilon)$. Then item 1 follows.

Item 2 follows from Example 6. Item 3 is proved directly from the definition. □

The formulas of the form $A \vee B$ or $A; B$ are not, in general, digitizable even with the condition that A and B are digitizable. For example, let $A \mathrel{\widehat{=}} \lceil P \rceil^*$, $B \mathrel{\widehat{=}} \lceil \neg P \rceil^*$, From Theorem 13, A, B are digitizable. Let $\mathcal{I}$ be as in Item 1 of Example 5. Then, for all $0 \leq \epsilon < 1$, $\mathcal{I}^\epsilon, [0, 4] \models_1 A \vee B$, $\mathcal{I}^\epsilon, [0, 4] \models_1 A; B$ However, $\mathcal{I}, [0, 4] \not\models A \vee B$ and $\mathcal{I}, [0, 4] \not\models A; B$.

Similar to the result in [2] about weak-closed formulas, we have:

Proposition 1. *Let A, B be digitizable DC formulas, k be an integer. Then the formula $(A \wedge \ell = k); B$ is digitizable.*

Example 7. Suppose we want to express that a signal sent on a wire P will be received after δ ($\delta \in \mathbb{N}$) time units on a wire Q. Using DC, this is written as a formula D:

$$D \mathrel{\widehat{=}} \lceil P \rceil; \ell = \delta \Rightarrow \ell = \delta; \lceil Q \rceil.$$

Since the formula $\ell = \delta; \lceil Q \rceil$ is not digitizable, we cannot expect that $\ell = \delta; \lceil Q \rceil$ is observed in the integer model of time. To make it observable, from Theorem 13 we should strengthen it by $\ell = \delta; (\lceil Q \rceil \wedge \ell \geq 1)$. Thus the premise should be weakened to $\lceil P \rceil \wedge \ell \geq 1; \ell = \delta$.

Hence, in order to implement the requirement in the integer model of time, we should weaken the requirement to:

$$D_1 \mathrel{\widehat{=}} (\lceil P \rceil \wedge \ell \geq 1); \ell = \delta \Rightarrow \ell = \delta; (\lceil Q \rceil \wedge \ell \geq 1) \ .$$

6 A Model of a Biphase Mark Protocol

In this section, as an example, we will use DC to model and prove the correctness of a Biphase Mark Protocol (BPM). The BPM protocol (or format) is widely used in practice for asynchronous communication between two digital hardware devices. The BPM protocol works by coding each bit of message as a portion of square wave (called a cell) of fixed clock cycles. Each cell is logically divided into two parts. The first part is called a mark, the other a code. If the signal in the mark subcell is the same as one in the code subcell then information carried by that cell is 0. Otherwise, if the signals are different then the information carried is 1. There is a phase reverse between two consecutive cells. It means that the signal at the beginning of the following cell is held as the negation of the signal at the end of the previous cell. Despite its extensive use, to the best of our knowledge, in the literature there are relatively few works dedicated to formally modeling and assessing the protocol's performance. In [3] the authors used a model of linear hybrid systems to analyze a protocol developed by Philips Corp. That protocol uses Manchester encoding that in many aspects is similar to a BPM protocol. In [11], Moore used the Boyer-Moore Logic to analyze a BPM protocol. In his version of the BPM protocols, each bit of message is coded as a cell of 18 clock cycles. Among them the 5 first cycles constitute the mark sub-cell and the last 13 cycles form the code sub-cell. He proved the correctness of the protocol provided the ratio of the clock rates of sender (or writer) and receiver (or reader) is within 5%. The author conjectures that his protocol works with clock rate ratio up to 30%. Inspired by Moore's work, we use DC to construct a model for the protocol. We adopt the same assumptions about the physical environment as in [11] and succeed in showing that his conjecture about 30% clock rate tolerance is the case.

We consider two digital devices called a writer and a reader. They are connected by a bus. The writer is required to send a message M which is a list of bits $< b_0, b_1, \ldots, b_n >$ to the reader. To do that, the writer holds the voltage in the bus at one of two values $\{low, high\}$. At the beginning, the writer holds voltage low for 18 cycles then it changes the voltage to high in order to transmit the first bit b_0. After 5 cycles, depending on the value of b_0 the writer keep voltage high for the next 13 cycles if $b_0 = 0$ or changes to low if $b_0 = 1$. For the next bit b_1 the writer changes the voltage of the mark subcell such that it is a negation of the voltage in code subcell of the previous cell. After transmission of bit b_n the writer keeps the voltage low for another 18 cycles to signal the end of session. For the reader's part, it first waits until detection of voltage change (or "edge") then skips a fixed number d of cycles (this number is called "sampling distance") and samples the signal there. It compares this signal with one after the edge. The reader assumes the bit of message is 1 if they are different or 0 otherwise.

The "raison d'etre" for such a protocol is the presence of disturbances from the physical environment on the communication process. In our model, we focus on two physical phenomena:

1. The voltage on the bus does not change immediately. It takes a significant amount of time to change from $high$ to low and vice versa.
2. The disparity between the clock rates of the writer and the reader.

As in [11] we adopt the following assumptions:

A1 On every cycle of its clock, the writer puts on the bus one of two values $\{low, high\}$. If the value of the current cycle is the same as the value of previous one then the signal on the bus stays stable at that value during the whole cycle.

A2 Otherwise, if those values are different then the signal on the bus is nondeterministic for the whole cycle i.e., the value for that whole cycle is either low or high.

These assumptions are abstractions from the real value of voltage on the bus. They allow us to see writer's and reader's views of bus as standard state variables while preserving the ability to model the temporal character of voltage change.

Our model is "normalized" to the writer's clock i.e., we assume that its clock rate is 1. The clock rate of the reader is r. Recall that we are modeling a BPM protocol in which the cell's size is $n = 18$, the mark subcell's size is $m = 5$, the code subcell's size is $k = 13$ and the sampling distance is $d = 10$. Without loss of generality, we assume further that the phase displacement (the fist tick) of the reader's clock is within the first cycle of the writer's clock.

We denote the writer's and reader's views by state variables X and Y respectively. Note that X can be discontinuous at integers only. We define a state variable C that has value 1 in code subcells and value 0 in mark subcells. Logically, the writer's view of the protocol can be considered as a chain of cells, so we could write for large enough T

$$\mathcal{I}, [0, T] \models St; ((\lceil \neg C \rceil \wedge \ell = 5); (\lceil C \rceil \wedge \ell = 13))^n; St \ , \tag{2}$$

where St is used to mark the start and the end of signal train. We use the convention that stipulates the existence of two intervals of length greater than the cell size in which X has low value. One interval lies prior to the first bit cell and the other after the last bit cell. So, $St \Rightarrow \lceil \neg X \rceil \wedge \ell \geq 18$. n is the length of message to be transmitted. We take the liberty of using A^n for $n-$times repetition of A. Assumptions about the protocol allow us to write down the following formulas:

$$\lceil C \rceil \Rightarrow (\lceil X \rceil \vee \lceil \neg X \rceil) \, , \tag{3}$$

$$\lceil \neg C \rceil \Rightarrow (\lceil X \rceil \vee \lceil \neg X \rceil) \, , \tag{4}$$

$$\lceil C \rceil; \lceil \neg C \rceil \Rightarrow (\lceil X \rceil; \lceil \neg X \rceil) \vee (\lceil \neg X \rceil; \lceil X \rceil) \ . \tag{5}$$

The formulas (3) and (4) say that during a subcell the signal writer holds is stable. (5) says that there is a signal edge to mark the start of next cell.

Assumption A1 imposes the following formulas on X and Y:

$$\lceil X \rceil \wedge \ell > 1 \Rightarrow \ell = 1; \lceil Y \rceil \tag{6}$$
$$\lceil \neg X \rceil \wedge \ell > 1 \Rightarrow \ell = 1; \lceil \neg Y \rceil \ . \tag{7}$$

Assumption A2 imposes the following formulas

$$(\lceil X \rceil \wedge \ell = 1); (\lceil \neg X \rceil \wedge \ell \leq 1) \Rightarrow \ell = 1; (\lceil Y \rceil \vee \lceil \neg Y \rceil) \tag{8}$$
$$(\lceil \neg X \rceil \wedge \ell = 1); (\lceil X \rceil \wedge \ell \leq 1) \Rightarrow \ell = 1; (\lceil Y \rceil \vee \lceil \neg Y \rceil) \ . \tag{9}$$

Note that the described assumptions do not impose any constraint on the possible value of Y during the first writer's clock cycle: we assume Y low for whole that period.

We also define two additional state variables M_X and M_Y on the basis of X and Y respectively as follows.

$$M_X(t) \mathrel{\widehat{=}} \begin{cases} 1 \text{ if } \exists t'.\ (0 < t - t' \leq 1) \wedge ((X([t', t)) = 0\ \wedge X([t' - q, t')) = 1) \vee \\ \quad (X([t', t)) = 1 \wedge\ X([t' - q, t')) = 0)) \\ 0 \text{ elsewhere,} \end{cases}$$

$$M_Y(t) \mathrel{\widehat{=}} \begin{cases} 1 \text{ if } \exists t'.\ (0 < t - t' \leq r) \wedge ((Y([t', t)) = 0\ \wedge Y([t' - qr, t')) = 1) \vee \\ \quad (Y([t', t)) = 1 \wedge\ Y([t' - qr, t')) = 0)) \\ 0 \text{ elsewhere,} \end{cases}$$

where $Z([a, b)) = v \ \mathrel{\widehat{=}}\ \forall t \in [a, b)\ Z(t) = v$ with Z a state variable and $v \in \{0, 1\}$. q is a constant that is often chosen to be half size of cell, in our case $q = 9$.

$M_X(t) = 1$ says that t is in the first cycle of a mark subcell (therefore of a cell) from the writer's view. On the other hand, the intuitive meaning of $M_Y(t) = 1$ is that t is a moment when the start of a cell can be detected by the reader and it commences to look for next bit. The stability of the signal in intervals $[t' - q, t')$ and $[t' - qr, t')$ is intended to distinguish an edge denoting the start of a cell from an edge that occurs inside a cell carrying a message bit 1. Fig. 2 illustrates the state variables in the case when the writer sends the message $< 1, 0 >$.

We are now in a position to formulate the properties that ensure the correctness of the BPM protocol. They are the following:

R1 Each bit sent by the writer is faithfully received by the reader by sampling with the sampling (clock rate) step r. The bits in the lists are ordered by time

R2 Each bit received by the reader corresponds to a bit sent by the writer.

In other words, requirement R1 establishes an injective mapping from the bits in the list sent by the writer to the bits in the list received by the reader while requirement R2 says that the mapping is surjective. R1 and R2 together ensure the mapping to be bijective.

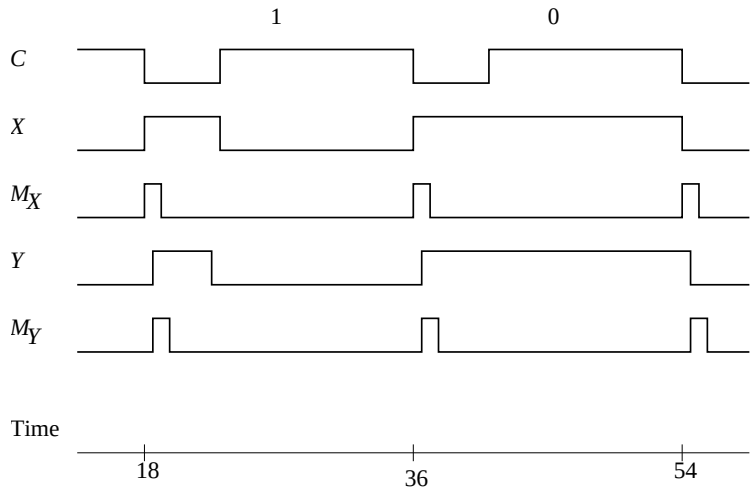

Fig. 2. States X, Y, M_X, M_Y when the message is $< 1, 0 >$

These properties can be formalized by following relationships between DC formulas. For all a, b, $a \leq b$,

$$\mathcal{I}_X, [a,b] \models \ell = 18 \wedge (\lceil M_X \rceil; true) \wedge \lceil X \rceil \text{ implies}$$
$$\mathcal{I}_Y, [a,b] \models_r \begin{pmatrix} \lceil \neg M_Y \rceil^*; (\lceil M_Y \rceil \wedge \lceil Y \rceil \wedge \ell = r); \\ \ell = 10r; (\lceil Y \rceil \wedge \ell = r); true \end{pmatrix} \tag{10}$$
$$\mathcal{I}_X, [a,b] \models \ell = 18 \wedge (\lceil M_X \rceil; true) \wedge \lceil \neg X \rceil \text{ implies}$$
$$\mathcal{I}_Y, [a,b] \models_r \begin{pmatrix} \lceil \neg M_Y \rceil^*; (\lceil M_Y \rceil \wedge \lceil \neg Y \rceil \wedge \ell = r); \\ \ell = 10r; (\lceil \neg Y \rceil \wedge \ell = r); true \end{pmatrix} \tag{11}$$
$$\mathcal{I}_X, [a,b] \models \begin{pmatrix} (\ell = 18 \wedge (\lceil M_X \rceil; true)) \wedge \\ ((\lceil X \rceil \wedge \ell = 5); (\lceil \neg X \rceil \wedge \ell = 13)) \end{pmatrix} \text{ implies}$$
$$\mathcal{I}_Y, [a,b] \models_r \begin{pmatrix} \lceil \neg M_Y \rceil^*; (\lceil M_Y \rceil \wedge \lceil Y \rceil \wedge \ell = r); \\ \ell = 10r; (\lceil \neg Y \rceil \wedge \ell = r); true \end{pmatrix} \tag{12}$$
$$\mathcal{I}_X, [a,b] \models \begin{pmatrix} (\ell = 18 \wedge \lceil M_X \rceil; true) \wedge \\ ((\lceil \neg X \rceil \wedge \ell = 5); (\lceil X \rceil \wedge \ell = 13)) \end{pmatrix} \text{ implies}$$
$$\mathcal{I}_Y, [a,b] \models_r \begin{pmatrix} \lceil \neg M_Y \rceil; (\lceil M_Y \rceil \wedge \lceil \neg Y \rceil \wedge \ell = r); \\ \ell = 10r; (\lceil Y \rceil \wedge \ell = r); true \end{pmatrix} \tag{13}$$
$$(\ true; (\lceil M_Y \rceil \wedge \ell < 1) \wedge \ell = 2 \Rightarrow \Diamond \lceil M_X \rceil \tag{14}$$
$$\lceil M_X \rceil \Rightarrow \ell \leq 1 \tag{15}$$
$$\lceil M_Y \rceil; \lceil \neg M_Y \rceil; \lceil M_Y \rceil \Rightarrow \ell \geq 5 \ . \tag{16}$$

Statements (10), (11), (12), (13) are formalization of requirement R1. In the premise parts, conjunctions of $\ell = 18$ and $\lceil M_X \rceil; true$ allow us to lock into an entire cell of the writer's view. The other conjuncts of the premises identify

whether the bit of message is 0 or 1. The conclusion parts in these statements effectively simulate the work of the reader. After detection of a mark, it skips 10 cycles and then samples at the next cycle. The bit is recovered by comparison of the sampled signal value with the one at the mark. So, (10) and (11) say that if the writer transmits 0 then the reader can recover 0, while (11) and (12) say the same thing for 1. It is in some sense the "local correctness". At a higher level, we can view whole transmission session as a series of cells connected by a chop operator. In total, (10) - (13) say that the all the bits sent by the writer are recovered by the reader in the same order. We are now concerned with the possibility that "superfluous" or "un-authentic" bits may sneak into the receiving list. However, such a possibility is excluded if the three formulas (14), (15), (16) are true. (14) establishes a mapping between a start of a cell in the reader's view represented by $\lceil M_Y \rceil$ to a start of a cell in the writer's view represented by $\lceil M_X \rceil$. We can imagine, in the time axis, intervals of length 2 where $\lceil M_Y \rceil$ holds in suffix parts. (14) says that in such intervals $\lceil M_X \rceil$ must hold somewhere. (15) and (16) in combination make that mapping injective. The truth of (16) implies that the mentioned intervals of length 2 can not overlap. Truth of (15) implies that $\lceil M_X \rceil$ in different intervals can not belong to the same cell of the writer's view.

Now we will discuss a subtle issue of different kinds of semantics that are involved in the proof of correctness of the BPM protocol. Certainly, the ultimate purpose of using the BPM protocol for communication is to recover information by a digital reader. Let us look at this process from the reader's point of view. The signal train denoted by state variable Y is all that is available. Because the asynchronous mode of communication is assumed, the reader does not have any relevant information about the source of that signal train nor the physical characteristics of the transmission medium. Therefore, it does not know about the properties of the signal train itself. What it can extract from "real" state variable Y is only a "digitized" version of Y. The latter is obtained using a digital clock of rate r with the first tick at arbitrary ϵ with respect to assumed real time. Then, information is recovered by the known algorithm on the basis of the "digitized" version of Y. There is no rich information about the relationship between the clocks of writer and reader except the assumption that the first tick of the reader's clock takes place during the first cycle of the writer's clock. So, to describe the writer's behavior from the reader position, continuous semantics should be used. Therefore, to prove statements $(10) - (13)$ that involve both continuous and sampling semantics, we have to use the trans-semantic technique investigated in previous sections. We proceed using the following thread of reasoning. First we will use

- the hypothesis that the clock rate ratio between the writer and the reader is within 30% , $H \ \widehat{=} \ 0.7 < r < 1.3$ together with
- the assumptions about the working mode of the writer - formulas $(2) - (5)$, and
- the assumptions about the physical characteristics of the communication medium - formulas $(6) - (9)$

to prove in standard DC the following lemma that gives continuous counterparts of $(10) - (13)$.

Lemma 3.

$$\begin{pmatrix} \ell = 18 \wedge \\ (\lceil M_X \rceil; true) \wedge \\ \lceil X \rceil \end{pmatrix} \Rightarrow \begin{pmatrix} \lceil \neg M_Y \rceil^* \ ; \\ (\lceil M_Y \rceil \wedge \lceil Y \rceil \wedge \ell = r) \ ; \\ \ell = 10r; (\lceil Y \rceil \wedge \ell = r); true \end{pmatrix} \quad (17)$$

$$\begin{pmatrix} \ell = 18 \wedge \\ \lceil M_X \rceil; true \wedge \\ \lceil \neg X \rceil \end{pmatrix} \Rightarrow \begin{pmatrix} \lceil \neg M_Y \rceil^* \ ; \\ (\lceil M_Y \rceil \wedge \lceil \neg Y \rceil \wedge \ell = r) \ ; \\ \ell = 10r; (\lceil \neg Y \rceil \wedge \ell = r); true \end{pmatrix} \quad (18)$$

$$\begin{pmatrix} \ell = 18 \ \wedge (\lceil M_X \rceil; true) \wedge \\ ((\lceil X \rceil \wedge \ell = 5) \ ; \\ (\lceil \neg X \rceil \wedge \ell = 13)) \end{pmatrix} \Rightarrow \begin{pmatrix} \lceil \neg M_Y \rceil^* \ ; \\ (\lceil M_Y \rceil \wedge \lceil Y \rceil \wedge \ell = r) \ ; \\ \ell = 10r; (\lceil \neg Y \rceil \wedge \ell = r); true \end{pmatrix} \quad (19)$$

$$\begin{pmatrix} \ell = 18 \wedge (\lceil M_X \rceil; true) \wedge \\ ((\lceil \neg X \rceil \wedge \ell = 5) \ ; \\ (\lceil X \rceil \wedge \ell = 13)) \end{pmatrix} \Rightarrow \begin{pmatrix} \lceil \neg M_Y \rceil^* \ ; \\ (\lceil M_Y \rceil \wedge \lceil \neg Y \rceil \wedge \ell = r) \ ; \\ \ell = 10r; (\lceil Y \rceil \wedge \ell = r); true \end{pmatrix} . \quad (20)$$

Then, the proof of statements $(10) - (13)$ con be carried out as follows. From the hypothesis of $(10) - (13)$ and Lemma 15, we conclude that the right hand side formulas in Lemma 15 are true (continuous semantics). Next, based on the results of the previous sections (Theorem 2, Corollary 3), we conclude the sampling semantics w.r.t. r of the right hand sides of the lemma from their syntax. Readers interested in the detailed proof of formulas $(17) - (20)$ are referred to [8].

Those who are familiar with [11] will notice the differences between his modeling and the one presented here. One of them is that in our modeling we do not treat phase displacement (the time moment of the first tick of the reader's clock) individually. Thanks to the power of explicitly expressing intervals offered in DC we are able to consider all possible phase displacements at once by using the ceiling operator as in $\lceil M_Y \rceil$, $\lceil M_X \rceil$. We think that this characteristic feature of DC also enables us to "squeeze" the model to get stronger propositions about the clock rate tolerance. In Moore's model "ramp" intervals in which signal is nondeterministic may be exaggerated through the three-pass process from the writer's view to the reader's view depending on possible values of the phase displacement and the rate of reader's clock. But in our model we use state variable D to express this property independently of the reader's clock. In conclusion, we think that by this example we have demonstrated the power of DC in dealing with problems that involve both continuous and discrete presentations of time.

7 Conclusion

We have treated the discretization of DC formulas by giving some discrete semantic models to DC. The best aspect we can see is the sampling semantics. It gives some ideas on how to derive a design from a specification, and to move closer to implementation in the same language.

We have also considered the digitizability of DC. In general, we cannot observe the truth of a formula that involves integrals of state expressions or the chop operator.

Since programming activities should generally be done in discrete time whereas true analog behavior should use continuous time, our treatments are useful for the designers of real-time systems from their specifications written in DC. It also shows that DC can be linked with other real-time logics for discrete time such as Interval Temporal Logics [12].

In this paper, we have used the semantic approach to discretizing DC. In our future work, we will use the syntactic approach to the problem so that one can use proof assistant tools in verifying the discretized systems.

References

1. P. J. Antsaklis, J. A. Stiver, M. Lemmon, Hybrid system modeling and autonomous control systems, LNCS 736, 1993, pp. 366-392.
2. A. Bouajjani, R. Echahed and R. Robbana, Verifying Invariance Properties of Timed Systems with Duration Variables, Formal Techniques in Real-Time and Fault-Tolerant Systems, LNCS 863, 1994, pp. 193-210.
3. D. Bosscher et al. Verification of an Audio Control Protocol, *LNCS 863*, 1994, pp. 170-192.
4. Z. Chaochen, C. A. R. Hoare, A. P. Ravn, A calculus of durations, *Information Processing Letters*, 40, 1992, pp. 269-276.
5. Z. Chaochen, A. P. Ravn, M. R. Hansen, Extended Duration Calculus for Hybrid Real-time Systems, LNCS 736, 1993, pp. 36-59.
6. Z. Chaochen, M.R. Hansen and P. Sestoft, Decidability and Undecidability Results for Duration Calculus, In *STACS'93*, P. Enjalbert, A. Finkel and K.W. Wagner (eds), LNCS 665, Springer-Verlag 1993, pp. 58-68.
7. M.R. Hansen, Model-Checking Discrete Duration Calculus *Formal Aspect of Computing*, 1994, pp. 826-845.
8. D.V. Hung and P.H. Giang, A Sampling Semantics of Duration Calculus, UNU/IIST report, No. 50, 1996.
9. D.V. Hung and W. Ji, On the design of hybrid control systems using I/O automata, UNU/IIST report, No. 34, 1994.
10. T. Henzinger, Z. Manna, and A. Pnueli, What Good are Digital Clocks?, Springer-Verlag, LNCS 623, 1992.
11. J. S. Moore A formal model of asynchronous communication and its use in mechanically verifying a Biphase Mark Protocol, *Formal Aspects of Computing*, 6, 1994, pp. 60-91.
12. B. Moszkowski, A temporal Logic for Multilevel Reasoning about Hardware, *IEEE Computer*, Vol. 18, No. 2, 1985, pp. 10-19.
13. A. Nerode, W. Kohn, Models for hybrid systems: automata, topologies, controllability, observability, LNCS 736, 1993, pp. 317-356.
14. X. Nicollin, A. Olivero, J. Sifakis, S. Yovine, An approach to the description and analysis of hybrid systems, LNCS 736, 1993, pp. 149-178.
15. A. Pnueli, Development of Hybrid Systems, Formal Techniques in Real-Time and Fault-Tolerant Systems, LNCS 863, 1994, pp. 77-8.