Gabbay Separation for the Duration Calculus

Dimitar P. Guelev ORCID
Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Sofia, Bulgaria
http://www.math.bas.bg/~gelevdp/

Abstract

Gabbay's separation theorem about linear temporal logic with past has proved to be one of the most useful theoretical results in temporal logic. In particular it enables a concise proof of Kamp's seminal expressive completeness theorem for LTL. In 2000, Alexander Rabinovich established an expressive completeness result for a subset of the Duration Calculus (DC), a real-time interval temporal logic. DC is based on the *chop* binary modality, which restricts access to subintervals of the reference time interval, and is therefore regarded as *introspective*. The considered subset of DC is known as the $\lceil P \rceil$-subset in the literature. Neighbourhood Logic (NL), a system closely related to DC, is based on the *neighbourhood* modalities, also written $\langle A \rangle$ and $\langle \bar{A} \rangle$ in the notation stemming from Allen's system of interval relations. These modalities are *expanding* as they allow writing future and past formulas to impose conditions outside the reference interval. This setting makes temporal separation relevant: is expressive power ultimately affected, if past constructs are not allowed in the scope of future ones, or vice versa? In this paper we establish an analogue of Gabbay's separation theorem for the $\lceil P \rceil$-subset of the extension of DC by the neighbourhood modalities, and the $\lceil P \rceil$-subset of the extension if DC by the neighbourhood modalities and *chop*-based analogue of *Kleene star*. We show that the result applies if the *weak chop inverses*, a pair *binary* expanding modalities are given the role of the neighbourhood modalities, by virtue of the inter-expressibility between them and the neighbourhood modalities in the presence of *chop*.

Keywords: Gabbay separation · Neighbourhood Logic · Duration Calculus · expanding modalities

2012 ACM Subject Classification Author: Please fill in 1 or more \ccsdesc macro

Keywords and phrases Gabbay separation · Duration Calculus · expanding modalities

Digital Object Identifier 10.4230/LIPIcs...

Introduction

Separation for Linear Temporal Logic (LTL, cf., e.g., [28]) was established by Dov Gabbay in [14]. Separation is about expressing temporal properties without making reference to the past in the scope of future constructs and vice versa. Gabbay proved that such a restriction does not affect the ultimate expressive power of past LTL, by a syntactically defined translation from arbitrary formulas to ones that are *separated*, i.e., satisfy the restriction. The applications of this theorem are numerous and important on their own right. They include a concise proof of Kamp's seminal expressive completeness result for LTL (see, e.g., [13]), the elimination of the past modalities from LTL, which simplifies the study of extensions of LTL, c.f., e.g., [10], Fisher's clausal normal form for past LTL [12], other normal forms [19, 15], etc. In this paper we establish an analogue of Gabbay's separation theorem for the extension of a subset of the Duration Calculus (DC) with a pair of expanding modalities known as the *neighbourhood modalities*, with and without the *chop*-based analogue of Kleene star, which is also called *iteration* in DC.

The Duration Calculus (DC, [33, 31]) is an extension of real time *Interval Temporal Logic* (ITL), which was first proposed by Moszkowski for discrete time [24, 25, 11]. DC is a real-time interval-based predicate logic for the modeling of hybrid systems. Unlike time points, time intervals, the possible worlds in DC, have an internal structure of subintervals. This justifies calling modalities like *chop* *introspective* for their providing access to these

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

subintervals only. Modalities for reaching outside the reference interval are called *expanding*. Several sets of such modalities have been proposed in the literature.

In this paper we prove a separation theorem for the $\lceil P \rceil$-subset of DC with the expanding *neighbourhood modalities* $\Diamond_l$ and $\Diamond_r$ added to DC's *chop* and *iteration*. The system based on $\Diamond_l$ and $\Diamond_r$ only, also written $\langle A \rangle$ and $\langle \overline{A} \rangle$ after Allen's interval relations [3], is called Neighbourhood Logic (NL, [4]) whereas we target DC with $\Diamond_l$ and $\Diamond_r$. Our theorem holds with *iteration* excluded too. We write DC-NL (DC-NL*) for DC with $\Diamond_l$ and $\Diamond_r$ (and *iteration*). In separated formulas, $\Diamond_d$ cannot not appear in the scope of other modalities, except $\Diamond_d$, $d = l, r$. $\Diamond_r$-free formulas are regarded as *past*, and $\Diamond_l$-free formulas are *future*. The *strict* forms of past (future) formulas are defined by further restricting *chop* and *iteration* to occur only in the scope of a $\Diamond_l$ ($\Diamond_r$). DC is a predicate logic. We prove that formulas in each of $\lceil P \rceil$-subsets of DC-NL and DC-NL* have *separated* equivalents in their respective subsets. These subsets are compatible with the system from Rabinovich's expressive completeness result [30]. We also show that the *weak chop inverses*, which are *binary* expanding modalities, are expressible using $\Diamond_l$ and $\Diamond_r$ in the considered subset. Their use in the *Mean-value Calculus*, another system from the DC family, was studied in [26]. $\Diamond_l$ and $\Diamond_r$ are definable using the weak chop inverses. Consequently, our separation theorem applies to the extensions of DC and DC* by the weak chop inverses too.

The technique of our proofs builds on our finds from [16] which led to establishing separation for discrete time ITL.

Structure of the paper: Section 1 gives preliminaries on DC and DC*, the weak chop inverses, and a supplementary result on quantification over state in DC. In Section 2 we state our separation theorem for the $\lceil P \rceil$-subsets of DC-NL and DC-NL* and give a simple example application. Section 3 is dedicated to the proof. The transformations for separating DC-NL and DC-NL* formulas are given in Sections 3.2 and 3.3, respectively, and use a lemma which is given in the preceding Section 3.1. Section 4 is about the expressibility of the weak chop inverses in the $\lceil P \rceil$-subsets of DC-NL and DC-NL*, using the lemma from Section 3.1 too. This implies that separation works for the extensions of DC and DC* by this pair of expanding modalities too. We conclude by pointing to some related work and making some comments on the relevance of the result.

1 Preliminaries

An in-depth presentation of DC and its extensions can be found in [31]. The syntax of the $\lceil P \rceil$-subset of DC is built starting from a set V of *state variables*. It includes *state expressions* S and *formulas* A. Let P stand for a *state variable*. The BNFs are:

$$S ::= \mathbf{0} \mid P \mid S \Rightarrow S \qquad\qquad A ::= \perp \mid \lceil\rceil \mid \lceil S \rceil \mid A \Rightarrow A \mid A;A$$

Semantics Given a set of state variables V, the type of *valuations* I is $V \times \mathbb{R} \to \{0,1\}$. Valuations I are required to have *finite variability*:

For any $P \in V$ and any bounded interval $[a,b] \subset \mathbb{R}$ there exists a finite sequence $t_0 = a < t_1 < \ldots < t_n = b$ such that $\lambda t.I(P,t)$ is constant in (t_{i-1}, t_i), $i = 1, \ldots, n$.

The *value* $I_t(S)$ *of state expression* S *at time* $t \in \mathbb{R}$ is defined by the clauses:

$$I_t(\mathbf{0}) \mathrel{\hat{=}} 0, \qquad I_t(P) \mathrel{\hat{=}} I(P,t), \qquad I_t(S_1 \Rightarrow S_2) \mathrel{\hat{=}} \max\{I_t(S_2), 1 - I_t(S_1)\}.$$

Satisfaction has the form $I,[a,b] \models A$, where $[a,b] \subset \mathbb{R}$. The defining clauses are:

$$\begin{array}{ll} I,[a,b] \not\models \bot, & I,[a,b] \models \lceil\rceil \ \text{ iff } \ a=b, \\ I,[a,b] \models \lceil S\rceil & \text{iff } \ a<b \text{ and } I_t(S)=1 \text{ for all but finitely many } t\in[a,b], \\ I,[a,b] \models A \Rightarrow B & \text{iff } \ I,[a,b]\models B \text{ or } I,[a,b]\not\models A, \\ I,[a,b] \models A;B & \text{iff } \ I,[a,m]\models A \text{ and } I,[m,b]\models B \text{ for some } m\in[a,b]. \end{array}$$

The connectives $\neg$, $\wedge$, $\vee$ and $\Leftrightarrow$ are defined as usual in both state expressions and formulas. Furthermore $\mathbf{1} \mathrel{\hat{=}} \mathbf{0} \Rightarrow \mathbf{0}$ and $\top \mathrel{\hat{=}} \bot \Rightarrow \bot$. A formula A is *valid* in DC, written $\models A$, if $I,[a,b] \models A$ for all I and all intervals $[a,b]$. In this paper we consider the extension of the $\lceil P\rceil$-subset of DC by the *neighbourhood modalities* $\diamond_d$, $d \in \{l,r\}$. The defining clauses for their semantics are as follows:

$$I,[a,b] \models \diamond_l A \text{ iff } I,[l,a] \models A \text{ for some } l \le a,\ I,[a,b] \models \diamond_r A \text{ iff } I,[b,r] \models A \text{ for some } r \ge b.$$

The universal duals $\Box_d$ of $\diamond_d$ are defined by putting $\Box_d A \mathrel{\hat{=}} \neg\diamond_d\neg A$, $d \in \{l,r\}$. *Chop* $A;B$ is written $A^\frown B$ in much of the literature. We write DC-NL for the extension of DC by $\diamond_l$ and $\diamond_r$. We also consider DC-NL*, the extension of DC-NL by *iteration*, the *chop*-based form of Kleene star, included. The defining clause for this operator is

$$\begin{array}{lll} I,[a,b] \models A^* & \text{iff} & a=b \text{ or there exist an increasing finite sequence } m_0=a<m_2<\cdots<m_n=b \\ & & \text{such that } I,[m_{i-1},m_i]\models A \text{ for } i=1,\ldots,n. \end{array}$$

Iteration is interdefinable with *positive iteration* $A^+ \mathrel{\hat{=}} A;(A^*)$, which we assume to be the derived one of the two: $\models A^* \Leftrightarrow \lceil\rceil \vee A^+$.

Predicate DC **and** NL include a (defined) flexible constant ℓ for the length $b-a$ of reference interval $[a,b]$. Using ℓ, *chop* can be defined in NL:

$$A;B \ \mathrel{\hat{=}}\ \exists x\exists y(x+y=\ell \wedge \diamond_l\diamond_r(A\wedge \ell=x)\wedge\diamond_r\diamond_l(B\wedge\ell=y)).$$

This definition is not available in NL's $\lceil P\rceil$-subset, hence the need to specify DC-NL.

Quantification over state in DC is defined by the clause

$$I,[a,b] \models \exists P\, A \text{ iff } I',[a,b] \models A \text{ for some } I' \text{ s. t. } I'(Q,t)=I(Q,t) \text{ and all } Q\in V\setminus\{P\},\ t\in\mathbb{R}.$$

Quantification over state is expressible in the $\lceil P\rceil$-subset of DC*:

▶ **Theorem 1.** *For every $\lceil P\rceil$-formula A in* DC* *and every state variable P there exists a (quantifier-free) $\lceil P\rceil$-formula B in* DC* *such that $\models B \Leftrightarrow \exists P\, A$.*

Mind that B is not guaranteed to be *iteration*-free, even in case A is.

This theorem follows from a correspondence between stutter-invariant regular languages and the $\lceil P\rceil$-subset that led to the decidability of the $\lceil P\rceil$-subset in [32]. It is not our contrubution, but the transformations from its proof supplement those from our other proofs.

Notation In this paper write ε, possibly with subscripts, to denote *optional* occurrences of the negation sign $\neg$, e.g, ε_Q below. We write $[A/B]C$ to denote the result of simultaneously replacing all the occurrences of B by A in C, e.g., $[\mathbf{0}/P]S$ below.

Proof of Theorem 1. Following [32], A translates into a regular expression over the alphabet

$$\Sigma \ \mathrel{\hat{=}}\ \{ \bigwedge_{Q \text{ is a state variable in } A} \varepsilon_Q Q : \varepsilon_Q \text{ is either } \neg \text{ or nothing}\}\,. \qquad (1)$$

The translation clauses are as follows:

$$t(\bot) \mathrel{\hat{=}} \emptyset \qquad t(\lceil S\rceil) \mathrel{\hat{=}} (\{\sigma \in \Sigma : \models \sigma \Rightarrow S\})^{+} \qquad t(A;B) \mathrel{\hat{=}} t(A);t(B)$$
$$t(\lceil\rceil) \mathrel{\hat{=}} \epsilon \text{ (the empty string)} \qquad t(A \Rightarrow B) \mathrel{\hat{=}} t(B) \cup \Sigma^* \setminus t(A) \qquad t(A^*) \mathrel{\hat{=}} t(A)^*$$

Up to equivalence, t can be inverted. Regular expressions admit complementation- and $\cap$-free equivalents; hence these operations can be omitted in the converse translation $\bar{t}$:

$$\bar{t}(\emptyset) \mathrel{\hat{=}} \bot \quad \bar{t}(a) \mathrel{\hat{=}} \lceil a\rceil \text{ for } a \in \Sigma \quad \bar{t}(R_1 \cup R_2) \mathrel{\hat{=}} \bar{t}(R_1) \vee \bar{t}(R_2) \quad \bar{t}(R^*) \mathrel{\hat{=}} \bar{t}(R)^*$$
$$\bar{t}(\varepsilon) \mathrel{\hat{=}} \lceil\rceil \quad \bar{t}(\Sigma^*) \mathrel{\hat{=}} \lceil\rceil \vee \lceil \mathbf{1}\rceil \quad \bar{t}(R_1;R_2) \mathrel{\hat{=}} \bar{t}(R_1);\bar{t}(R_2)$$

Given a regular expression $R = t(A)$, $\bar{t}(R')$ is equivalent to A for any R' that defines the same language as R. Applying $\bar{t}$ to a complementation- and $\cap$-free equivalent R' to $t(A)$ produces an equivalent to A with $\vee$ as the only propositional connective, except possibly inside state expressions. Given this, $\exists P$ can be eliminated from formulas of the form $\bar{t}(R')$:

$$\models \exists P\, \bot \Leftrightarrow \bot \quad \models \exists P\, \lceil S\rceil \Leftrightarrow \lceil [\mathbf{0}/P]S \vee [\mathbf{1}/P]S\rceil^{+} \quad \models \exists P\,(A_1;A_2) \Leftrightarrow \exists P\, A_1; \exists P\, A_2$$
$$\models \exists P\, \lceil\rceil \Leftrightarrow \lceil\rceil \quad \models \exists P\,(A_1 \vee A_2) \Leftrightarrow \exists P\, A_1 \vee \exists P\, A_2 \quad \models \exists P\, A^* \Leftrightarrow (\exists P\, A)^*.$$

The equivalence $\exists P\, \lceil S\rceil$ above hinges on the finite variability of $I_t(P)$. ◀

The weak chop inverses A/B and $A\backslash B$, cf., e.g., [26], are defined by the clauses:

$$I,[a,b] \models A/B \quad \text{iff} \quad \text{for all } r \geq b, \text{ if } I,[b,r] \models B \text{ then } I,[a,r] \models A.$$
$$I,[a,b] \models A\backslash B \quad \text{iff} \quad \text{for all } l \leq a, \text{ if } I,[l,a] \models B \text{ then } I,[l,b] \models A.$$

$\Diamond_l A$ and $\Diamond_r A$ can be defined as $\neg(\bot\backslash A)$ and $\neg(\bot/A)$, respectively. In Section 4 we show how A/B and $A\backslash B$ can be expressed using $\Diamond_l$ and $\Diamond_r$ too for $\lceil P\rceil$-formulas A and B, but with the expressing formulas built in a more complex way.

Separation as Known for LTL We relate the setting and statement of Gabbay's separation theorem about past LTL as our work builds in the example of this theorem. Let p stand for an atomic proposition. Discrete time LTL formulas with past have the syntax:

$$A ::= \bot \mid p \mid A \Rightarrow A \mid \bigcirc A \mid A\,\mathcal{U}\,A \mid \ominus A \mid A\,\mathcal{S}\,A$$

$\ominus$ and $\mathcal{S}$ are the past mirror operators of $\bigcirc$ and $\mathcal{U}$. $\ominus$- and $\mathcal{S}$-free formulas are called *future* formulas, and $\bigcirc$- and $\mathcal{U}$-free formulas are called *past*. Formulas of the form $\bigcirc F$ where F is future are called *strictly future*. In [14], Dov Gabbay demonstrated that any formula in LTL with past is equivalent to a Boolean combination of past and strictly future formulas for flows of time which are either finite or infinite, in either the future or the past, or both.

Modal heights $h_{\Diamond_l}(.)$, $h_{\Diamond_r}$ and $h_{*}(.)$ of formulas wrt the neighbourhood modalities and *iteration*, aka Kleene star appear in our inductive reasoning below. In general, $h(A)$ denotes the length of the longest chain of A's subformulas, including A itself, with the main connective being the specified modality wrt the (transitive closure of) the subformula relation.

2 The Separation Theorem

In this section we formulate the main contrubution of the paper, Theorems 2 and 3, which is a separation theorem for the $\lceil P\rceil$-subsets of DC-NL and DC-NL*, and use the theorem to demonstrate the expressibility of an interval-based version of the 'past-forgetting' operator from [18] as a simple example application.

We call DC-NL (DC-NL*) formula F (non-strictly) *future* if it has the syntax

$$F ::= C \mid \neg F \mid F \vee F \mid \Diamond_r F$$

where C stands for a DC (DC*) formula, where *chop* and *iteration* are the only modalities. Non-strictly *past* formulas are defined similarly, with $\Diamond_l$ instead of $\Diamond_r$. A *separated formula* is a Boolean combination of past and future formulas.

Following the example of LTL, we call Boolean combinations of $\Diamond_l$-, resp. $\Diamond_r$-formulas with non-strict past, resp. future operands *strictly past*, resp. *strictly future* formulas. Such formulas can impose no conditions on the reference interval; they only refer to the adjacent past and future parts of the timeline. These adjacent parts still include the respective endpoints of the reference interval. However the $\lceil P \rceil$ construct cannot discern interpretations I of the state variables such that $\lambda t.I(P,t)$ differ at finitely many time points only. Unlike that, in discrete time an extra step away from the present time using $\ominus$, resp., $\bigcirc$ is necessary to prevent a formula from imposing conditions on the reference time point or a reference interval's endpoint. The shared time point 'prevents' *chop* of discrete time ITL from being a *separating conjunction* in the sense of [29], whereas DC *chop* meets the requirements. *Separated* formulas are Boolean combinations of strictly past formulas, strictly future formulas and *introspective* (just DC*) formulas, where the only modalities are *chop* and *iteration*, that are known as *introspective* too.

▶ **Theorem 2.** *Let A be a $\lceil P \rceil$-formula in* DC-NL *(*DC-NL**). Then there exists a separated $\lceil P \rceil$-formula A' in* DC-NL *(*DC-NL**) such that $\models A \Leftrightarrow A'$.*

In Section 4 we demonstrate the inter-expressibility between $(./.)$ and $(.\backslash.)$, and $\Diamond_l$ and $\Diamond_r$, respectively. This implies that Theorem septhmmain holds for the weak chop inverses instead of the respective $\Diamond_d$, $d \in \{l, r\}$ too:

▶ **Theorem 3.** *Let A be a $\lceil P \rceil$-formula in the extension of* DC *(*DC**) by $(./.)$ and $(.\backslash.)$. Then there exists a separated $\lceil P \rceil$-formula A' in* DC *(*DC**) such that $\models A \Leftrightarrow A'$.*

An Example Application: Expressing the N operator The temporal operator N ('now') was proposed for past LTL in [18], see also [17], as a means for preventing 'access' into the past beyond the time of applying N. Assuming $\sigma \mathrel{\hat=} \sigma^0\sigma^1\ldots$ to be a sequence of states

$$\sigma, i \models_{\mathrm{LTL}} \mathsf{N}A \text{ iff } \sigma^i\sigma^{i+1}\ldots, 0 \models_{\mathrm{LTL}} A\ .$$

If an arbitrary closed interval $D \subseteq \mathbb{R}$, and not only the whole of $\mathbb{R}$, is allowed to be the time domain, N can be defined for (real-time) DC-NL too. With such time domains, the endpoints of 'all time' can be identified, because, e.g., $D, I, [a,b] \models \Box_l \lceil\rceil$ iff $a = \min D$. (Since the $\lceil P \rceil$-subset of DC-NL is merely *topological*, as opposed to *metric*, it cannot distinguish *open* time domains from $\mathbb{R}$.) We can define N on intervals by putting:

$$\begin{array}{l} D, I, [a,b] \models \mathsf{N}_l A \text{ iff } \{x \in D : x \geq a\}, I, [a,b] \models A \\ D, I, [a,b] \models \mathsf{N}_r A \text{ iff } \{x \in D : x \leq b\}, I, [a,b] \models A \end{array}$$

Theorem 2 entails that N_l and N_r are expressible in DC-NL:

▶ **Proposition 4.** DC-NL $+\, \mathsf{N}_l, \mathsf{N}_r$ *has the same expressive power as* DC-NL.

Proof. Let A' be a separated equivalent of A. Then $\models \mathsf{N}_d A \Leftrightarrow [\Diamond_d(B \wedge \lceil\rceil)/\Diamond_d B : B \in \mathsf{Subf}(A')]A'$, $d \in \{l, r\}$. ◀

3 The Proof of Separation for DC-NL and DC-NL*

In this section we propose a set of valid equivalences which, if appropriately used as transformation rules starting from some arbitrary given formula from the $\lceil P\rceil$-subset of DC-NL*, lead to a separated formula in DC-NL*. If the given formula is *iteration*-free, i.e., in DC-NL, then so is the separated equivalent. This amounts to proving Theorem 2.

Our key observation is that formulas which are supposed to be evaluated at intervals that extend some given interval into either the future or the past have equivalents which consist of subformulas to be evaluated at the given interval and subformulas to be evaluated at intervals which are adjacent to it, these two subintervals being appropriately referenced using *chop* as parts of the enveloping interval. In our proof of separation, this observation is refered to as a lemma that states the possibility to express any introspective formula as a case distinction of *chop*-formulas with the LHS (RHS) operands of chop forming a full system. The lemma can be seen as a generalization of the *guarded normal form*, which is ubiquitous in process logics, with the full systems of guards describing a primitive opening move replaced by full systems of interval-based temporal conditions to be satisfied at whatever prefixes (suffixes) of the reference runs necessary. Later on we use the lemma in expressing $(./.)$ $((.\backslash.))$ in terms of $\Diamond_r$ $(\Diamond_l)$ too.

3.1 The Key Lemma

A finite set of formulas $A_1,\dots,A_n$ is a *full system*, if $\models \bigvee\limits_{k=1}^{n} A_k$ and, given $1\le k_1<k_2\le n$, $\models \neg(A_{k_1}\wedge A_{k_2})$.

▶ **Lemma 5.** *Let A be a $\lceil P\rceil$-formula in* DC *(*DC**). Then there exists an $n<\omega$ and some* DC *(*DC**) $\lceil P\rceil$-formulas A_k, A'_k, $k=1,\dots,n$, such that $A_1,\dots,A_n$ form a full system and*

$$\models A \Leftrightarrow \bigvee_{k=1}^{n} A_k;A'_k \ \text{ and } \ \models A \Leftrightarrow \bigwedge_{k=1}^{n} \neg(A_k;\neg A'_k). \tag{2}$$

Furthermore, $h_*(A_k)\le h_*(A)$ and $h_*(A'_k)\le h_*(A)$.

Informally, this means that, $I,[a,b]\models A$ iff whenever $m\in[a,b]$ and $I,[a,m]\models A_k$, $I,[m,b]\models A'_k$ holds. Furthermore, for every $m\in[a,b]$ there is a unique k such that $I,[a,m]\models A_k$. Interestingly, the construct $\neg(F;\neg G)$ used in the second equivalence (2) is regarded as a form of *temporal implication*, written $F\mapsto G$, in ITL [23, 5]. This construct is akin to *suffix implication* [2], see also [1]. It requires the suffix of an interval to satisfy B, if the complementing prefix satisfies A. Much like $\Rightarrow$'s being the right adjoint of $\wedge$, $\mapsto$ is the right adjoint of *chop*:

$$\models A\mapsto(B\mapsto C)\Leftrightarrow(A;B)\mapsto C\ .$$

Since *chop* is a *separating conjunction* in DC, $\mapsto$ also fits the description of the corresponding *bunched implication* [29]. In this paper we stick to the notation in terms of *chop* for both $\mapsto$ and its mirror $\neg(\neg G;F)$.

Proof of Lemma 5. Induction on the construction of A. For $\bot$, $\lceil\rceil$ and $\lceil P\rceil$, we have:

$$\bot\ \Leftrightarrow\ (\top;\bot)\quad \lceil\rceil\ \Leftrightarrow\ (\lceil\rceil;\lceil\rceil)\vee(\neg\lceil\rceil;\bot)\quad \lceil P\rceil\ \Leftrightarrow\ (\lceil P\rceil;(\lceil P\rceil\vee\lceil\rceil))\vee(\lceil\rceil;\lceil P\rceil)\vee(\neg(\lceil\rceil\vee\lceil P\rceil);\bot)$$

Let $B_1,\ldots,B_n$, $B'_1,\ldots,B'_n$ satisfy 2 for B and $C_1,\ldots,C_m$, $C'_1,\ldots,C'_m$ satisfy 2 for C. Then:

$$\begin{array}{rcl} B\, op\, C & \Leftrightarrow & \bigvee\limits_{k=1}^{n}\bigvee\limits_{l=1}^{m}(B_k\wedge C_l;(B'_k\, op\, C'_l)),\ \ op\in\{\Rightarrow,\vee,\wedge,\Leftrightarrow\} \\ B;C & \Leftrightarrow & \bigvee\limits_{k=1}^{n}\ \bigvee\limits_{X\subseteq\{1,\ldots,m\}}\left(B_k\wedge\bigwedge\limits_{l\in X}(B;C_l)\wedge\bigwedge\limits_{l\not\in X}\neg(B;C_l)\right);\left((B'_k;C)\vee\bigvee\limits_{l\in X}C'_l\right) \end{array}$$

For the equivalence about *iteration*, let $C \mathrel{\hat{=}} B\vee\lceil\rceil$ and $C_1,\ldots,C_m$, $C'_1,\ldots,C'_m$ be as above. Then $B^*\Leftrightarrow C^*$, and:

$$B^* \quad \Leftrightarrow \quad \bigvee\limits_{X\subseteq\{1,\ldots,m\}}\left(\bigwedge\limits_{l\in X}(B^*;C_l)\wedge\bigwedge\limits_{l\not\in X}\neg(B^*;C_l)\right);\left(\bigvee\limits_{l\in X}(C'_l;B^*)\right)$$

The equivalences on the right in (2) are written similarly. The RHSs of these equivalences have the form required in the lemma. Using these equivalences as transformation rules bottom up, an arbitrary A can be given that form.

A direct check is sufficient for establishing (2) about $\bot$, $\lceil\rceil$ and $\lceil P\rceil$. The case of $B\, op\, C$, esp. $op = \Rightarrow$, admits the proof that works for the *Guarded Normal Form* in [6].

For the equivalence on the left in (2) about $B;C$, ($\Rightarrow$), let $I,[a,b]\models B;C$, $t,m\in[a,b]$, and $I,[a,m]\models B$ and $I,[m,b]\models C$. Assuming $I,[a,b]\models B;C$, if $t\in[a,m]$, then $I,[a,m]\models B_k$ for some unique k. If $t\in[m,b]$, then a unique $X\subseteq\{1,\ldots,m\}$ exists such that $I,[a,m]\models B;C_l$ holds iff $l\in X$. The conjunctions of $B_k\wedge\bigwedge\limits_{l\in X}(B;C_l)\wedge\bigwedge\limits_{l\not\in X}\neg(B;C_l)$, $k=1,\ldots,n$, $X\subseteq\{1,\ldots,m\}$ form a full system because so do both the B_ks, and the conjunctions $\bigwedge\limits_{l\in X}(B;C_l)\wedge\bigwedge\limits_{l\not\in X}\neg(B;C_l)$, $X\subseteq\{1,\ldots,m\}$. Since $I,[a,m]\models B$ and $I,[m,b]\models C$, for an $[a,t]$ satisfying the member of this full system for any given k and X, we can conclude that $I,[t,b]\models(B'_k;C)\vee\bigvee\limits_{l\in X}C'_l$ from the assumptions on the B'_ks and the C'_ls. For the converse implication ($\Leftarrow$), let $[a,b]$ be an arbitrary interval, $t\in[a,b]$, and $I,[a,t]\models B_k\wedge\bigwedge\limits_{l\in X}(B;C_l)\wedge\bigwedge\limits_{l\not\in X}\neg(B;C_l)$, which is bound to be true for some unique pair k,X. Then, $I,[t,b]\models B'_k;C$ implies $I,[a,b]\models B_k;B'_k;C$, and $I,[m,b]\models C'_l$ implies $I,[a,b]\models B;C_l;C'_l$ for any $l\in X$. In both cases $I,[a,b]\models B;C$ follows because $\models B_k;B'_k\Rightarrow B$ and $\models C_l;C'_l\Rightarrow C$. The $\Leftarrow$ direction similarly follows from $\models B_k;B'_k\Rightarrow B$ and $\models C\Rightarrow C_l;C'_l$ for some appropriately chosen k and l. The LHS equivalence (2) about B^* is established similarly, with the use of C facilitating a uniform handling of the case of B^* holding trivially at 0-length intervals. The RHS equivalences (2) follow from the LHS ones by the assumption that the A_ks form a full system.

Observe that this equivalence satisfies $h_*(B_k)\leq h_*(B)$ and $h_*(B'_k)\leq h_*(B)$, where B_k and B'_k can be identified from the syntax of the RHS. The non-increase of $h_*(.)$ can be checked directly for the equivalences which do not feature *iteration* explicitly too, but may nevertheless become used for processing formulas with *iteration*. This implies $h_*(A_k)\leq h_*(A)$ and $h_*(A'_k)\leq h_*(A)$. ◀

The time mirror image of Lemma 5 holds too, with the time mirror of (2) reading

$$\models A\Leftrightarrow\bigvee_{k=1}^{n}A'_k;A_k \text{ and } \models A\Leftrightarrow\bigwedge_{k=1}^{n}\neg(\neg A'_k;A_k).$$

The proof is no different because all the modalities are symmetrical wrt the direction of time. For this reason, in the sequel we omit ‘mirror’ statements and their proofs.

On the complexity of the transformations from Lemma 5. Interestingly, a peak (exponential) blowup in the transformations from Lemma 5's proof occurs in the clause for *chop* and not the clause for $\neg$, the typical source of such blowups. However, a closer look at the inductive assumptions shows that the pairwise inconsistency achieved at the cost of using $A_k \wedge \bigwedge\limits_{l\in X}(A;B_l) \wedge \bigwedge\limits_{l\notin X} \neg(A;B_l)$ for all $k \in \{1,\dots,m\}$ and the 2^n different $X \subseteq \{1,\dots,m\}$ in the required full system is instrumental for the correctness of the clause about the binary Boolean connectives, where negation is obtained for $op = \Rightarrow$ and $B = \bot$. Hence this blowup can be linked to the alternation of $\neg$ and monotone operators such as *chop* that is common in proofs of the non-elementariness of the blowup upon reaching normal forms.

Lemma 5 admits an automata-theoretic proof, along the lines of the proof of Theorem 1. We have sketched such a proof for discrete time ITL in [16]. That proof leads to different A_k and A'_k satisfying (2) for the same A, and allows a non-elementary upper bound on the length of these formulas to be established using the size of a deterministic FSM recognizing A. Unlike the automata-based proof, the equivalences of this proof suggest transformations that are valuable for their compositionality and their validity in DC in general, and not just for the $\lceil P \rceil$-subset. Furthermore, the proof given here facilitates establishing that *-height is not increased upon moving to the RHSs of (2).

3.2 Separating the Neighbourhood Modalities in DC-NL and DC-NL*

In this section we prove Theorem 2 by showing how occurrences of $\Diamond_d$ can be taken out of the scope of *chop*, $\Diamond_{\overline{d}}$, $d \in \{l,r\}$, $\overline{l} \mathrel{\hat=} r$, $\overline{r} \mathrel{\hat=} l$ and *iteration*. The transformations that we propose are supposed to be applied bottom up, on formulas with *chop*, *iteration* or $\Diamond_d$, $d \in \{l,r\}$, as the main connective, and assuming that the operands of these connectives are already separated. If the main connective is $\Diamond_d$, then we need to target only the $\Diamond_{\overline{d}}$-subformulas in $\Diamond_d$'s operand, possibly at the cost of introducing some $\Diamond_{\overline{d}}$-subformulas in the scope of *chop*, to be subsequently extracted from there too. If the main connective is *chop* or *iteration*, then separation requires extracting the occurrences of $\Diamond_d$ for both $d = l$ and $d = r$.

To show that the above transformations combine into a terminating procedure which produces a separated formula, for DC-NL, we reason by induction on the $\Diamond_d$-height of the relevant formulas. In the case of DC-NL*, we also keep track of *-height, which is not increased upon applying Lemma 5, nor by the transformations for separating formulas with $\Diamond_l$, $\Diamond_r$ or *chop* as the main connective, but can be increased upon eliminating an 'intermediate' appearance of a quantification over state by an application of Theorem 1. The use of such a quantification in the course of transformations, and the subtle observations on the quantified formulas which enable the conclusion that this potential increase of *-height is unrelated to termination become clear in due course below. In most of the cases, we give detail only on the extracting of $\Diamond_r$-subformulas, because of the time symmetry.

Separating $\Diamond_d$-formulas Let $d = l$; the case of $d = r$ is its mirror. Since

$$\models \Diamond_l(A_1 \vee A_2) \Leftrightarrow \Diamond_l A_1 \vee \Diamond_l A_2 \ , \tag{3}$$

the availability of DNF for A of $\Diamond_l A$ makes it sufficient to consider the case of A of the form $P \wedge \bigwedge\limits_{k=1}^{n} \varepsilon_k \Diamond_r F_k$ where P is (non-strictly) past and $F_1,\dots,F_n$ are future. Observe that

$$\models \Diamond_l\left(P \wedge \bigwedge_{k=1}^{n} \varepsilon_k \Diamond_r F_k\right) \Leftrightarrow \Diamond_l P \wedge \bigwedge_{k=1}^{n}((\lceil\rceil \wedge \varepsilon_k \Diamond_r F_k);\top)\ . \tag{4}$$

Transforming formulas according to (3) and (4) does not change $\Diamond_r$-height but implies that finding a separated equivalent to $\Diamond_l A$ boils down to separating $((\lceil\rceil \wedge \varepsilon\Diamond_r F_k);\top)$, which are the *chop*-formulas. Here follow the transformations for doing this.

Separating *chop*-formulas We need to consider only *chop* applied to conjunctions of introspective formulas and possibly negated past $\Diamond_l$-formulas or future $\Diamond_r$-formulas because

$$\models (L_1 \vee L_2); R \Leftrightarrow (L_1; R) \vee (L_2; R) \text{ and } L; (R_1 \vee R_2) \Leftrightarrow (L; R_1) \vee (L; R_2)$$

Here '*past*' ('*future*') restricts the operands of $\Diamond_l$ ($\Diamond_r$), making the formulas *strictly* past (future). Such formulas can be extracted from the left (right) operand of *chop* using that

$$\models (L \wedge \varepsilon\Diamond_l P); R \Leftrightarrow (L; R) \wedge \varepsilon\Diamond_l P \text{ and } \models L; (R \wedge \varepsilon\Diamond_r F) \Leftrightarrow (L; R) \wedge \varepsilon\Diamond_r F. \quad (5)$$

Much like (3), this does not affect $\Diamond_d$-height. It remains to consider $(L \wedge \bigwedge_{k=1}^{n} \varepsilon_k \Diamond_r F_k); R$, which, by virtue of the time symmetry, explains $L; (R \wedge \bigwedge_{k=1}^{n} \varepsilon_k \Diamond_l P_k)$ too.

The transformations of formulas of the form $(L \wedge \varepsilon\Diamond_r F); R$ below are about the designated $\varepsilon\Diamond_r F$ only, and are supposed to be used repeatedly, if L has more conjuncts of this form. Transformations which extract designated $\Diamond_r F$s ($\Diamond_l P$s) from $(L \wedge \varepsilon\Diamond_r F); R$ ($L; (R \wedge \varepsilon\Diamond_r P)$) can be applied in any order with no obstructive interaction occurring.

$(L \wedge \Diamond_r F); R$: By (3), F can be assumed to be a conjunction $C \wedge G$ where C is introspective and G is strictly future. Let C_k, C'_k, $k = 1, \ldots, n$, satisfy Lemma 5 for C. We can use that

$$\models (L \wedge \Diamond_r(C \wedge G)); R \Leftrightarrow (L; (R \wedge ((C \wedge G); \top))) \vee \bigvee_{k=1}^{n} (L; (R \wedge C_k)) \wedge \Diamond_r(C'_k \wedge G)$$

and further process the RHS of $\Leftrightarrow$ in it. The two disjuncts on the RHS above correspond to F being satisfied at an interval which is shorter, or the same length, or longer than the one which presumably satisfies R. Since C_k and C'_k are introspective, the newly introduced formulas $\Diamond_r(C'_k \wedge G)$ on the RHS of $\Leftrightarrow$ are separated. $(L; (R \wedge (C \wedge G; \top)))$ can be separated too because $h_{\Diamond_r}(G) < h_{\Diamond_r}((L \wedge \Diamond_r F); R)$.

$(L \wedge \neg\Diamond_r F); R$: Then by the distributivity (3) of $\Diamond_r$ over $\vee$ again, $\neg F$ can be assumed to have the form $C \vee G$ where C and G are like in the case of a non-negated $\Diamond_r$-subformula. Satisfying $(L \wedge \neg\Diamond_r \neg(C \vee G)); R$ requires $(C \vee G)$ to hold at all the intervals which start at the right end of the one where L presumably holds. Therefore we can use that

$$\models (L \wedge \Box_r(C \vee G)); R \Leftrightarrow \bigvee_{k=1}^{n} L; (R \wedge C_k \wedge \neg(\neg(C \vee G); \top)) \wedge \Box_r(C'_k \vee G)$$

where $\Box_r \,\hat{=}\, \neg\Diamond_r\neg$. Again, the RHS of that equivalence has a strictly future G to be further extracted from the left operand of the newly introduced $(\neg(C \vee G); \top)$. This can be accomplished because $h_{\Diamond_r}(G) < h_{\Diamond_r}((L \wedge \neg\Diamond_r F); R)$. Finally, whatever $\Diamond_r$-subformulas happen to occur the separated equivalent of $(\neg(C \vee G); \top)$, can be extracted from the *chop* where they appear in the right operand using (5).

The transformations above are sufficient for establishing Theorem 2 about DC-NL. By Lemma 5, these transformations do not cause *-height to increase. This is relevant in separating formulas in DC-NL*, which is explained next.

3.3 Separating *iteration* formulas

To extract $\Diamond_l$ and $\Diamond_r$ from the scope of *iteration*, we use the inter-expressibility between *iteration* and quantification over state, and the expressibility of quantification over state in the $\lceil P \rceil$-subset of DC* (Theorem 1). Let B be some $H_1 \vee \ldots \vee H_q$ where H_p, $p = 1, \ldots, q$, is a conjunction of introspective formulas and possibly negated past $\Diamond_l$-formulas and future $\Diamond_r$-formulas. This form of B can be achieved because B is assumed to be separated upon considering the separation of B^*. Furthermore, the operands of the past $\Diamond_l$-conjuncts (future $\Diamond_r$-conjuncts) in $H_1, \ldots, H_q$ can be assumed to be conjunctions of introspective and strictly past (future) formulas, because of (3).

We introduce the state variables $T, S_1, \ldots, S_q$ and first replace B^* by the RHS of the valid equivalence

$$\left(\bigvee_{p=1}^{q} H_p\right)^* \Leftrightarrow \lceil\rceil \vee \exists T \exists S_1 \ldots \exists S_q \left((\lceil T \rceil; \lceil \neg T \rceil) \wedge \bigvee_{p=1}^{q} (\lceil S_p \rceil \wedge H_p)\right)^+ . \tag{6}$$

in which, if $a < b$, $I, [a,b] \models B^*$ is stated to be equivalent to the existence of a partition of $[a,b]$ into a finite sequence of maximal $\lceil T \rceil; \lceil \neg T \rceil$-intervals $[m_0, m_1], \ldots, [m_{n-1}, m_n]$ where $m_0 = a < m_1 < \ldots < m_n = b$, with each of these intervals also satisfying some of $\lceil S_1 \rceil, \ldots, \lceil S_q \rceil$ and the corresponding H_p, $p = 1, \ldots, q$. In the context of $T, S_1, \ldots, S_q$ satisfying this condition, any future conjunct $\varepsilon \Diamond_r F$ of H_j must hold at the intervals $[m_{i-1}, m_i]$, $i = 1, \ldots, n$, where $\lceil S_j \rceil$ holds. The relevant m_k can be identified by the conditions that $S_p \wedge \neg T$ holds in a left neighbourhood of m_i, and, for $i < n$, T holds in a right neighbourhood of m_i. If $I, [m_{i-1}, m_i] \models H_j$, then, depending on ε, either $I, [m_i, z] \models F$ is required for *some* $z \geq m_i$ or $I, [m_i, z] \models \neg F$ is required for *all* $z \geq m_i$. The extraction of $\varepsilon \Diamond_r F$ can be achieved by 'deleting' $\varepsilon \Diamond_r F$ from H_j and 'inserting' a dedicated conjunct *outside* the $(.)^+$ of (6) to state that εF holds at the relevant $[m_i, z]$. To write this new conjunct for a (non-negated) $\Diamond_r F$, observe that, because of (3), F can be written as the conjunction $C \wedge G$ of some introspective formula C and some strictly future formula G. Furthermore, let C_k, C'_k, $k = 1, \ldots, m$, satisfy (2) for C. Then the conjunct in question can be written as

$$\alpha(F,j) \mathrel{\hat{=}} \left(\begin{array}{l} ((\top; \lceil S_j \rceil) \Rightarrow \Diamond_r (C \wedge G)) \wedge \\ \bigwedge\limits_{k=1}^{m} (\top; \lceil S_j \wedge \neg T \rceil; ((\lceil T \rceil; \top) \wedge \neg (C \wedge G; \top) \wedge C_k)) \Rightarrow \Diamond_r (C'_k \wedge G) \end{array}\right)$$

If ε is $\neg$, then, assuming $\bar{C}_k, \bar{C}'_k$, $k = 1, \ldots, \overline{m}$, to satisfy (2) for $\neg C$, the conjunct in question can be written as

$$\beta(F,j) \mathrel{\hat{=}} \left(\begin{array}{l} ((\top; \lceil S_j \rceil) \Rightarrow \neg \Diamond_r (C \wedge G)) \wedge \\ \neg (\top; \lceil S_j \wedge \neg T \rceil; ((\lceil T \rceil; \top) \wedge ((C \wedge G); \top))) \wedge \\ \bigwedge\limits_{k=1}^{\overline{m}} (\top; \lceil S_j \wedge \neg T \rceil; ((\lceil T \rceil; \top) \wedge \bar{C}_k)) \Rightarrow \Box_r (\bar{C}'_k \vee \neg G) \ . \end{array}\right)$$

Let $\gamma(\varepsilon, F, j)$ stand for $\beta(F,j)$, if $\varepsilon = \neg$, and $\alpha(F,j)$, otherwise. Then

$$\begin{array}{l} \models \left((\lceil T \rceil; \lceil \neg T \rceil) \wedge \left((\lceil S_j \rceil \wedge K \wedge \varepsilon \Diamond_r F) \vee \bigvee\limits_{\substack{p=1 \\ p \neq j}}^{q} (\lceil S_p \rceil \wedge H_p)\right)\right)^+ \Leftrightarrow \\ \qquad \left((\lceil T \rceil; \lceil \neg T \rceil) \wedge \left((\lceil S_j \rceil \wedge K) \vee \bigvee\limits_{\substack{p=1 \\ p \neq j}}^{q} (\lceil S_p \rceil \wedge H_p)\right)\right)^+ \wedge \gamma(\varepsilon, F, j). \end{array} \tag{7}$$

Extracting more conjuncts of the form $\varepsilon \Diamond_r F$ from (what is left of) $H_1, \ldots, H_q$, can be continued by similarly processing the RHS of (7). The occurrences of G, which is strictly future, in the left operands of *chop* in $\alpha(F,j)$ and $\beta(F,j)$ need to be extracted too. This can be done because $h_{\Diamond_r}(G) < h_{\Diamond_r}(\Diamond_r F)$. Past $\Diamond_l$-conjuncts can be extracted similarly, using the time mirrors of (6), $\gamma(\varepsilon, F, j)$ and (7). The repeated use of (7) and its and time mirror eventually lead to an introspective

$$([T];[\neg T]) \wedge \bigvee_{p=1}^{q} ([S_p] \wedge H_p)$$

in the scope $(.)^+$, which concludes the extraction of the expanding formulas from the scope of *iteration*. Completing the transformations requires eliminating the $\exists T \exists S_1 \ldots \exists S_q$ introduced in (6) too. Observe that the $\Diamond_l$- and $\Diamond_r$-subformulas which appear in the instances of $\gamma(\varepsilon, F, j)$ introduced inside the scope of $\exists T \exists S_1 \ldots \exists S_q$ have no occurrences of T, nor $S_1, \ldots, S_q$, and are linked with the remaining introspective subformulas, which may have such occurrences, by Boolean connectives only. Hence these $\Diamond_l$- or $\Diamond_r$-subformulas can be taken out of the scope of $\exists T \exists S_1 \ldots \exists S_q$ using the De Morgan laws and

$$\models \exists S\,(X \vee Y) \Leftrightarrow \exists S\, X \vee \exists S\, Y, \text{ and, for } S\text{-free } Z, \models \exists S(X \wedge Z) \Leftrightarrow Z \wedge \exists S\, X,$$

This means that Theorem 1, which is about introspective formulas only, applies, and $\exists T \exists S_1 \ldots \exists S_q$ can be eliminated. Hence Theorem 2 holds about DC-NL* too.

4 Expressing the Weak Chop Inverses by the Neighbourhood Modalities and Separation for the Weak Chop Inverses

In this section we prove that the weak chop inverses are expressible in DC-NL, which means that separation applies to DC with these expanding modalities instead of $\Diamond_l$ and $\Diamond_r$ too. This means that Theorem 3 follows from Theorem 2.

Suppose that A_1, A_2, B are separated formulas in DC-NL (DC-NL*). Then the availability of conjunctive normal forms and the validity of the equivalences

$$(A_1 \wedge A_2)/B \Leftrightarrow A_1/B \wedge A_2/B$$

entails that we need to consider only formulas A/B where A is a disjunction of introspective formulas and future and past formulas. Past disjuncts P in the 1st operand of $(./.)$ can be extracted using the validity of

$$(A \vee P)/B \Leftrightarrow P \vee A/B.$$

The following proposition shows how to express A/B in case A is a disjunction of introspective and possibly negated $\Diamond_r$-formulas.

▶ **Proposition 6.** *Let A be a $[P]$-formula in DC (DC*) and A_k, A'_k, $k = 1, \ldots, n$ satisfy Lemma 5 for A. Let B be a $[P]$-formula in DC-NL*. Let F be a conjunction of possibly negated $\Diamond_r$-formulas. Then*

$$\models (A \vee F)/B \Leftrightarrow \bigvee_{k=1}^{n} A_k \wedge \Box_r(B \Rightarrow (A'_k \vee F)) \ . \tag{8}$$

Proof. ($\Rightarrow$): Let $I, [a,b]$ satisfy the RHS of (8). Consider an arbitrary $r \geq b$ such that $I,[b,r] \models B$. Since there is a (unique) $k \in \{1,\ldots,n\}$ such that $I,[a,b] \models A_k$. We have $I,[a,r] \models A \vee F$ because $I,[b,r] \models A'_k \vee F$ and $\models A_k; A'_k \Rightarrow A$ by Lemma 5. The ($\Leftarrow$) direction is trivial to check and we omit it. ◀

The formula for A/B in terms of $\Diamond_l$ and $\Diamond_r$ in the RHS of (8) can be further separated to extract past subformulas of B from the scope of $\Box_r$ as in DC-NL (DC-NL*). The above argument shows that (./.)-formulas whose operands are in the $\lceil P\rceil$-subset of DC-NL (DC-NL*) have equivalents in the $\lceil P\rceil$-subset of DC-NL (DC-NL*) themselves. Observe that, in the presence of *chop*, it takes only $\Diamond_r$ to eliminate (./.). Similarly, (.\\.), which is about looking to the left of reference interval, can be eliminated using only *chop* and $\Diamond_l$. As mentioned in the Preliminaries section, expressing $\Diamond_l$ and $\Diamond_r$ by means of (.\\.) and (./.) is straightforward. This concludes our reduction of the $\lceil P\rceil$-subset of DC-NL (DC-NL*) with the weak chop inverses to the $\lceil P\rceil$-subset of DC-NL (DC-NL*), and entails that separation applies to that system too as stated in Theorem 3.

Concluding Remarks

In this paper we have shown how separation after Gabbay applies to the $\lceil P\rceil$-subsets of DC-NL and DC-NL*, the extensions of DC by the neighbourhood modalities. These subsets correspond to the subset of DC whose expressive completeness was demonstrated in [30].

The $\lceil S\rceil$-construct, which is definitive for the $\lceil P\rceil$-subsets of DC-NL and DC-NL*, has a considerable similarity with the *homogeneity principle* which is known from studies on neighbourhood logics of discrete time. That principle was proposed in [22, 20] and was adopted in a number of more recent works such as [7, 8, 9]. Unlike the *locality principle* from Moszkowski's (standard) discrete time ITL, where the satisfaction of an atomic proposition p is determined by the labeling of the initial state of the reference interval, homogeneity means that atomic proposition p must label all the states in the reference interval for p to hold at that interval as a formula. The two variants are ultimately interdefinable, but facilitate applications in a slightly different way. Homogeneity can be compared with DC's $\lceil P\rceil$ because $\lceil P\rceil$ means that P is supposed to hold 'almost everywhere' in the reference interval. The main difference is that varying valuations at a single point interval is negligible in real-time NL and DC, whereas the labeling of the single point in such an interval can be referred to in discrete time. This makes the difference between DC's *chop* being a *separating conjunction* [29] and ITL's *chop* not fitting that description. It is known that past expanding modalities increase the ultimate expressive power of discrete time ITL [21], and not just its succinctness, the latter being the case in past LTL. This adds to the relevance of algorithmic methods for interval-based expanding modalities in general.

Providing a separation theorem to the $\lceil P\rceil$-subset of DC-NL improves our understanding of the logic and may facilitate further results. One obvious avenue of future study would be to consider interval-based variants of the applications of separation that are known about point-based past LTL. In particular, one rather straightforward application would be to simplify the theoretical considerations that are needed for the study of extensions, especially branching time ones such as [27], by making it sufficient to consider future-only formulas, while still enjoying the succinctness contributed by the availability of past operators.

References

1. IEEE 1364-2005 - IEEE Standard for Verilog Hardware Description Language, 2005. URL: `https://ieeexplore.ieee.org/document/1620780`.
2. IEEE 1850-2010 - IEEE Standard for Property Specification Language (PSL), 2010. URL: `https://standards.ieee.org/standard/1850-2010.html`.
3. James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983. URL: `http://doi.acm.org/10.1145/182.358434`, `doi:10.1145/182.358434`.

4 Rana Barua, Suman Roy, and Zhou Chaochen. Completeness of neighbourhood logic. In Christoph Meinel and Sophie Tison, editors, *STACS 99, Proceedings*, volume 1563 of *LNCS*, pages 521–530. Springer, 1999. doi:10.1007/3-540-49116-3_49.

5 Marc Boulé and Zeljko Zilic. Psl and sva assertion languages. In *Generating Hardware Assertion Checkers: For Hardware Verification, Emulation, Post-Fabrication Debugging and On-Line Monitoring*, pages 55–82. Springer Netherlands, Dordrecht, 2008. doi:10.1007/978-1-4020-8586-4_4.

6 Howard Bowman and Simon Thompson. A Decision Procedure and Complete Axiomatisation of Finite Interval Temporal Logic with Projection. *Journal of Logic and Computation*, 13(2):195–239, 2003.

7 Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Interval vs. point temporal logic model checking: an expressiveness comparison. In Akash Lal, S. Akshay, Saket Saurabh, and Sandeep Sen, editors, *FSTTCS 2016*, volume 65 of *LIPIcs*, pages 26:1–26:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.26.

8 Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Interval vs. point temporal logic model checking: An expressiveness comparison. *ACM Trans. Comput. Log.*, 20(1):4:1–4:31, 2019. doi:10.1145/3281028.

9 Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Pietro Sala. Which fragments of the interval temporal logic HS are tractable in model checking? *Theor. Comput. Sci.*, 764:125–144, 2019. doi:10.1016/j.tcs.2018.04.011.

10 Laura Bozzelli, Aniello Murano, and Loredana Sorrentino. Alternating-time temporal logics with linear past. *Theor. Comput. Sci.*, 813:199–217, 2020. doi:10.1016/j.tcs.2019.11.028.

11 Antonio Cau, Ben Moszkowski, and Hussein Zedan. ITL web pages. URL: http://www.antonio-cau.co.uk/ITL/.

12 Michael Fisher. A normal form for temporal logics and its applications in theorem-proving and execution. *J. Log. Comput.*, 7(4):429–456, 1997. doi:10.1093/logcom/7.4.429.

13 Dov Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects. Volume I.* Oxford University Press, 1994.

14 Dov M. Gabbay. Declarative Past and Imperative Future: Executable Temporal Logic for Interactive Systems. In *Proceedings of the Colloquium of Temporal Logic in Specification*, volume 398 of *LNCS*, pages 67–89. Springer, 1989.

15 Dimitar P. Guelev. A syntactical proof of the canonical reactivity form for past linear temporal logic. *J. Log. Comput.*, 18(4):615–623, 2008. doi:10.1093/logcom/exn002.

16 Dimitar P. Guelev and Ben Moszkowski. A separation theorem for discrete-time interval temporal logic. *Journal of Applied Non-Classical Logics*, 32(1):28–54, 2022. doi:10.1080/11663081.2022.2050135.

17 François Laroussinie, Nicolas Markey, and Philippe Schnoebelen. Temporal logic with forgettable past. In *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*, pages 383–392. IEEE Computer Society, 2002. doi:10.1109/LICS.2002.1029846.

18 François Laroussinie and Philippe Schnoebelen. A hierarchy of temporal logics with past (extended abstract). In Patrice Enjalbert, Ernst W. Mayr, and Klaus W. Wagner, editors, *STACS 94, Proceedings*, volume 775 of *LNCS*, pages 47–58. Springer, 1994. doi:10.1007/3-540-57785-8_130.

19 Zohar Manna and Amir Pnueli. A Hierarchy of Temporal Properties. In *9th Symposium on Principles of Distributed Computing*, pages 377–408. ACM Press, 1990.

20 Alberto Molinari, Angelo Montanari, Aniello Murano, Giuseppe Perelli, and Adriano Peron. Checking interval properties of computations. *Acta Informatica*, 53(6-8):587–619, 2016. doi:10.1007/s00236-015-0250-1.

21 Dario Della Monica, Angelo Montanari, and Pietro Sala. The importance of the past in interval temporal logics: The case of propositional neighborhood logic. In Alexander Artikis, Robert

Craven, Nihan Kesim Cicekli, Babak Sadighi, and Kostas Stathis, editors, *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*, volume 7360 of *LNCS*, pages 79–102. Springer, 2012. doi:10.1007/978-3-642-29414-3_6.

22 Angelo Montanari, Aniello Murano, Giuseppe Perelli, and Adriano Peron. Checking interval properties of computations. In Amedeo Cesta, Carlo Combi, and François Laroussinie, editors, *Proceedings of TIME 2014*, pages 59–68. IEEE Computer Society, 2014. doi:10.1109/TIME.2014.24.

23 Ben Moszkowski. *Reasoning about Digital Circuits*. Ph.D. thesis, Department of Computer Science, Stanford University, 1983. URL: http://www.antonio-cau.co.uk/ITL/publications/reports/thesis-ben.pdf.

24 Ben Moszkowski. Temporal Logic For Multilevel Reasoning About Hardware. *IEEE Computer*, 18(2):10–19, 1985.

25 Ben Moszkowski. *Executing Temporal Logic Programs*. Cambridge University Press, 1986. URL: http://www.antonio-cau.co.uk/ITL/publications/reports/tempura-book.pdf.

26 Paritosh K. Pandya. Some extensions to propositional mean-value caculus: Expressiveness and decidability. In Hans Kleine Büning, editor, *CSL '95, Selected Papers*, volume 1092 of *LNCS*, pages 434–451. Springer, 1995. doi:10.1007/3-540-61377-3_52.

27 Paritosh K. Pandya. Model checking CTL*[DC]. In Tiziana Margaria and Wang Yi, editors, *TACAS 2001, Proceedings*, volume 2031 of *LNCS*, pages 559–573. Springer, 2001. doi:10.1007/3-540-45319-9_38.

28 Amir Pnueli. The Temporal Logic of Programs. In *Proceedings of the 18th IEEE Symposium Foundations of Computer Science*, pages 46–57. IEEE, 1977.

29 David J. Pym. *The semantics and proof theory of the logic of bunched implications*, volume 26 of *Applied logic series*. Kluwer, 2002.

30 Alexander Moshe Rabinovich. Expressive completeness of duration calculus. *Inf. Comput.*, 156(1-2):320–344, 2000. doi:10.1006/inco.1999.2816.

31 Zhou Chaochen and Michael R. Hansen. *Duration Calculus. A Formal Approach to Real-Time Systems*. Springer, 2004.

32 Zhou Chaochen, Michael R. Hansen, and Peter Sestoft. Decidability and undecidability results for duration calculus. In Patrice Enjalbert, Alain Finkel, and Klaus W. Wagner, editors, *STACS 93, Proceedings*, volume 665 of *LNCS*, pages 58–68. Springer, 1993. doi:10.1007/3-540-56503-5_8.

33 Zhou Chaochen, C. A. R. Hoare, and Anders P. Ravn. A Calculus of Durations. *Information Processing Letters*, 40(5):269–276, 1991.