

# Expressive completeness for LTL with modulo counting and group quantifiers

A V Sreejith <sup>1</sup>

*Institute of Mathematical Sciences,  
CIT Campus, Taramani,  
Chennai, India*

---

## Abstract

Kamp showed that linear temporal logic is expressively complete for first order logic over words. We give a Gabbay style proof to show that linear temporal logic extended with modulo counting and group quantifiers (introduced by Baziramwabo, McKenzie, Thérien) is expressively complete for first order logic with modulo counting (introduced by Straubing, Thérien, Thomas) and group quantifiers (introduced by Barrington, Immerman, Straubing).

*Keywords:* linear temporal logic, first order logic, modulo counting, group quantifiers

---

## 1 Introduction

Kamp showed that over linear orders, linear temporal logic LTL (Prior [Pri56], Pnueli [Pnu77], Gabbay, Pnueli, Shelah and Stain [GPSS80]) is expressively complete for first order logic FO with monadic predicates. Kamp's proof ([Kam68]) also establishes that FO has the three-variable property: every first order formula has an equivalent formula using just three variables. Gabbay [Gab87] gave another proof of this result emphasizing the separation property: every LTL formula has an equivalent formula which can be separated into pure past, present and pure future formulas. A more precise definition will appear later in this article. For word models, McNaughton and Papert [RS71] showed that these logics also correspond to the class of starfree regular languages. The translation algorithms are not elementary, and from Meyer and Stockmeyer [SM73] we know we cannot do better.

---

<sup>1</sup> Email: [sreejith@imsc.res.in](mailto:sreejith@imsc.res.in)

Various extensions have been proposed to extend the expressiveness of temporal logic over word models to all the regular languages (Wolper [Wol83], Banieqbal and Barringer [BB87], Hodkinson [Hod95], Baziramwabo and McKenzie and Thérien [BMT99], Henriksen and Thiagarajan [HT99]). Hodkinson’s proof (see the book ([Hus01])) represents finite automata using LTL with fixpoint operators, nowadays called the linear mu-calculus. The later work of Janin and Walukiewicz ([JW96]), which is in a branching context, also passes through automata.

Barrington, Immerman and Straubing [BIS90] showed using the Krohn-Rhodes decomposition of finite monoids that the regular languages can also be described by FOGRP, first order logic extended with generalized quantifiers which perform computation in finite groups. This generalizes FOMOD, first order logic extended with modulo counting, defined by Straubing, Thérien and Thomas ([STT95]). Baziramwabo, McKenzie, Thérien [BMT99] use the Krohn-Rhodes decomposition to produce an LTLGRP formula, LTL extended with group computation modalities. So they also establish the three-variable property for FOGRP. They also show that a suitable LTLMOD is expressively complete for FOMOD.

In this article we give Gabbay-style [Gab87] separation-based proofs of these latter expressive completeness results, showing that LTLGRP and LTLMOD have the separation property. As a technical note, the present definition of the logics LTLGRP and LTLMOD uses simpler modalities than the ones in [BMT99] and in our own earlier work ([LS10]).

We also look at unary temporal logic UTL ([EVW02]), which is LTL without the Until operator. We extend the technique of Etessami, Vardi and Wilke ([EVW02]) to show that UTL with group operators (UTLGRP) has the same expressive power as FOGRP which uses only two variables. The translation is exponential. We would like the referees to note that this last result is claimed in a paper under submission (with the proofs delegated to an appendix) and used to prove complexity results for FOGRP.

The motivation for our work is to push logic-based methods further. In our view an interesting open question remaining is whether one can “logicize” the result of Barrington, Immerman and Straubing [BIS90], namely, translate from monadic second order logic MSO to FOGRP without passing through automata or monoids.

Section 2 provides the syntax and semantics of LTL and first order logic with its extensions. Section 3 give a proof of our main theorem.

## 2 Preliminaries

### 2.1 Linear temporal logic

A *linear temporal logic* formula over a set of propositions  $\mathcal{P}$  is built using the following syntax

$$\alpha ::= p \in \mathcal{P} \mid \phi \mid \neg \alpha \mid \alpha \vee \beta \mid \mathbf{X}\alpha \mid \mathbf{Y}\alpha \mid \mathbf{F}\alpha \mid \mathbf{P}\alpha \mid \alpha \mathbf{U}\beta \mid \alpha \mathbf{S}\beta$$

Restricting the above logic to not have since (**S**) and until (**U**) operator, gives us the logic UTL. We denote by *true* the statement  $a \vee \neg a$ . Then  $\mathbf{F}\alpha$ ,  $\mathbf{P}\alpha$  can be defined as  $\text{true}\mathbf{U}\alpha$  and  $\text{true}\mathbf{S}\alpha$  respectively.

Linear temporal logic (LTL) formulas are interpreted on strings over an alphabet  $\Sigma = 2^{\mathcal{P}}$ , the set of all subsets of  $\mathcal{P}$ . We denote the  $i^{\text{th}}$  letter of a string  $u \in \Sigma^*$  by  $u(i)$  and the length of  $u$  by  $|u|$ . The positions in  $u$  are numbered from 0 to  $|u| - 1$ . We consider only finite words.

Given a string  $u \in \Sigma^*$ , a position  $i < |u|$  and a linear temporal logic formula  $\phi$  over  $\mathcal{P}$ , we denote by  $(u, i) \models \phi$  to mean that  $\phi$  is true at position  $i$  in the word  $u$ . The *semantics* of the logic is given below

$$\begin{aligned} (u, i) &\models p \text{ if } p \in u(i) \text{ and } p \in \mathcal{P} \\ (u, i) &\models \neg\alpha \text{ if not } (u, i) \models \alpha \\ (u, i) &\models \alpha \vee \beta \text{ if } (u, i) \models \alpha \text{ or } (u, i) \models \beta \\ (u, i) &\models \mathbf{X}\alpha \text{ if } i < |u| - 1 \text{ and } (u, i + 1) \models \alpha \\ (u, i) &\models \mathbf{Y}\alpha \text{ if } i > 0 \text{ and } (u, i - 1) \models \alpha \\ (u, i) &\models \alpha \mathbf{U}\beta \text{ if there exists a } j > i \text{ and } (u, j) \models \beta \text{ and for all } k, \\ &\quad \text{if } i < k < j \text{ implies } (u, k) \models \alpha \\ (u, i) &\models \alpha \mathbf{S}\beta \text{ if there exists a } j < i \text{ and } (u, j) \models \beta \text{ and for all } k, \\ &\quad \text{if } j < k < i \text{ implies } (u, k) \models \alpha \end{aligned}$$

As usual  $\mathbf{G}\alpha$  abbreviates  $\neg\mathbf{F}\neg\alpha$  and  $\mathbf{H}\alpha$  abbreviates  $\neg\mathbf{P}\neg\alpha$ .

We say that a formula  $\phi$  *satisfies* a word  $u$  if  $(u, 0) \models \phi$ . Then  $u$  is called a *model* of  $\phi$ . We denote by  $L(\phi)$  the language of  $\phi$ , that is the set of all the models of  $\phi$ . We say that formulas  $\alpha$  and  $\beta$  are *equivalent* if for all words  $w$  and  $\forall i < |w|$ , we have that  $w, i \models \alpha \Leftrightarrow w, i \models \beta$ .

## 2.2 Modulo counting and Group operator extensions of LTL

The logic LTLMOD extends LTL with the following modulo counting quantifiers for every  $r, q \in \mathbb{N}$  where  $r \leq q$ :

$$MOD_{r,q}^F \alpha \mid MOD_{r,q}^P \alpha$$

The following logic LTLGRP, extends LTL with the group operator.

$$G_{g,(g_1,\dots,g_k)}^F \langle \phi_1, \dots, \phi_k \rangle \mid G_{g,(g_1,\dots,g_k)}^P \langle \phi_1, \dots, \phi_k \rangle$$

Here  $G$  is a finite group whose elements are ordered  $g_1, \dots, g_k$  and  $g$  is an element in  $G$ . We denote by multiplication the group operator. If the ordering of the group is known we will drop that from the subscript of the syntax. Also we denote by LTLGRP( $G$ ) the logic if the only group used is  $G$ .

We denote by UTLMOD, UTLGRP the logic got by extending UTL with the above modulo counting and group operators.

Let  $\alpha$  be a formula over the propositions  $\mathcal{P}$  and  $u$  be a finite word over  $(2^{\mathcal{P}})^*$  and let  $i < |u|$ . The semantics for the newly introduced operators are given as follows.

$$u, i \models MOD_{r,q}^F \alpha \text{ iff } |\{i < l < |u| \mid u, l \models \alpha\}| \equiv_q r$$

$$w, i \models MOD_{r,q}^P \alpha \text{ iff } |\{0 \leq l < i \mid w, l \models \alpha\}| \equiv_q r$$

We denote by  $m \equiv_q r$  that  $m$  leaves a remainder of  $r$  when divided by  $q$ . The semantics for the group operators  $G_g^F \langle \phi_1, \dots, \phi_k \rangle$  is as follows. First, we define  $\Gamma(w, l) = g_j$  if  $w, l \models \neg \alpha_1 \wedge \dots \wedge \neg \alpha_{j-1} \wedge \alpha_j$  for  $1 \leq j \leq k$ . Also define  $\Gamma(w, l) = 1$  (the identity element) if none of the formulae  $\alpha_1, \dots, \alpha_k$  hold at position  $l$ . Then:

$$w, i \models G_g^F \langle \phi_1, \dots, \phi_k \rangle \text{ iff } (\prod_{l=i+1}^{|w|-1} \Gamma(w, l)) = g$$

$$w, i \models G_g^P \langle \phi_1, \dots, \phi_k \rangle \text{ iff } (\prod_{l=0}^{i-1} \Gamma(w, l)) = g$$

Observe that this is a generalization of the modulo counting we did earlier, since modulo counting is similar to working with cyclic groups. Later we use the notation  $\Gamma_j \langle \phi_1, \dots, \phi_k \rangle$  to denote the formula  $\neg \alpha_1 \wedge \dots \wedge \neg \alpha_{j-1} \wedge \alpha_j$

We say that an LTLGRP (LTLMOD) formula is a *pure future* formula if the only modalities used are  $X, U$  and future group operator (future mod operator). Similarly we say that an LTLGRP (LTLMOD) formula is a *pure past* formula if the only modalities used are  $Y, S$  and past group operator (past mod operator). *Pure present* formulas are those which does not use any modality. The formula  $F MOD_{r,q}^P \alpha$  is neither a pure past or a pure present or a pure future formula. In such a case we call the formula *impure*. Observe that the semantics of until,

since, group and modulo operators are strict, that is it does not depend on the present position.

We define the *future depth* (*past depth*) inductively. All pure past (future) formula has future (past) depth 0. Future depth of the formulas  $\phi_1 \mathbf{U} \phi_2$ ,  $\mathbf{X} \phi_1$ ,  $G_g^F \langle \phi_1, \dots, \phi_k \rangle$  is one more than the future depth of the formulas  $\phi_1, \dots, \phi_k$ . Similarly the past depth of the formulas  $\phi_1 \mathbf{S} \phi_2$ ,  $\mathbf{Y} \phi_1$ ,  $G_g^P \langle \phi_1, \dots, \phi_k \rangle$  is one more than the past depth of the formulas  $\phi_1, \dots, \phi_k$ .

The depth of a formula is the sum of its future depth and past depth. Similarly the *alternation* depth of a formula is the number of alternations of its future and past modalities.

### 2.3 First order logic with modulo and group quantifiers

Let  $\Sigma$  be a finite alphabet and  $\mathcal{V}$  be a finite set of variables. A *word model* over  $(\Sigma, \mathcal{V})$  is a pair  $(w, s)$ , where  $w \in \Sigma^*$  and  $s : \mathcal{V} \rightarrow \{1, \dots, m\}$ , where  $|w| = m$ . Let us introduce the syntax of first order logic for word models. We use a finite set of variables  $\mathcal{V} = \{x_1, \dots\}$ , a binary predicate  $<$  and unary predicates  $a$ ,  $a \in \Sigma$ , for the finite alphabet  $\Sigma$ .

$$\alpha ::= a(x_i), a \in \Sigma \mid x_1 < x_2 \mid x_1 = x_2 \mid \neg \alpha \mid \alpha \vee \beta \mid \exists x \alpha$$

The semantics for first order logic is standard. In the logic FOMOD, introduced by Straubing, Therien and Thomas [STT95], one can count the number of times a formula is satisfied modulo a number:

$$\exists^{r,q} x \phi, q \in \mathbb{N}, r \leq q$$

The semantics for  $\exists^{r,q} x \phi$  is given as follows.

$$(w, s) \models \exists^{r,q} x \phi \text{ iff } |\{l \in \{1, \dots, |w|\} \mid (w, s[x \mapsto l]) \models \phi\}| \equiv (r \pmod{q})$$

Here  $s[x \mapsto l]$  denotes extension of  $s$  with the variable  $x$  being mapped to  $l$ .

The logic FOGRP is got by extending FO with the following group quantifier [BIS90]:

$$G_{g,(g_1,\dots,g_k)} x \langle \phi_1, \dots, \phi_k \rangle$$

Its semantics is given as follows.

$$w, s \models G_{g,(g_1,\dots,g_k)} x \langle \phi_1, \dots, \phi_k \rangle \text{ iff } \prod_{i=1}^{|w|} \Gamma(w, s[x \mapsto i]) = g$$

The definition of  $\Gamma$  is the same as the one given in LTLGRP semantics.

The following theorem [Kam68,Gab87] identifies the set of languages accepted by LTL.

**Theorem 2.1** *LTL is expressively complete for First order logic over words.*

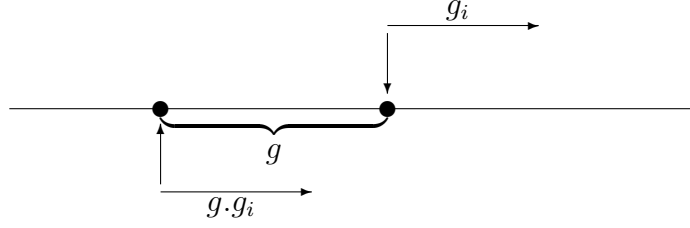


Fig. 1. Future Group modality can simulate Past Group modality

The theorem states that for all first order logic formula  $\phi(x)$  with one free variable, there exists a formula  $\psi$  in LTL such that for all words  $w$  and  $\forall i \leq |w|$  we have that

$$w, i \models \phi \Leftrightarrow w, i \models \psi$$

We show in the next section that the theorem continues to be true even if we add the power of modulo counting and group operations.

### 3 Properties of LTLgrp

We first look at certain properties of the logic LTLGRP. Our first observation is that the formulas in LTLGRP have a normal form.

**Theorem 3.1** *Let  $\alpha \in \text{LTLGRP}$ . Then there exists a group,  $G_\alpha$  and a formula  $\hat{\alpha} \in \text{LTLGRP}(G_\alpha)$  such that  $L(\alpha) = L(\hat{\alpha})$ .*

**Proof.** Take  $G_\alpha$  to be the cross product of all the groups in  $\alpha$ . Now any group  $G$  in  $\alpha$  can be replaced by  $G_\alpha$ .  $\square$

Hence we can always work with formulas over  $\text{LTLGRP}(G)$ , for some group  $G$ . Our next theorem says that the future group operator can simulate the past group operator (and vice versa) in the presence of future and past operators.

**Theorem 3.2** *Let  $\alpha \in \text{LTLGRP}$ . Then  $\alpha$  is equivalent to a formula  $\hat{\alpha}$ , where  $\hat{\alpha}$  do not contain any past group operator.*

**Proof.** We replace all past group operators by future group operators as follows. Let  $\beta := G_g^P \langle \phi_1, \dots, \phi_k \rangle$  be a formula. We assume that all the formulas  $\phi_1, \dots, \phi_k$  do not contain any past group operator. We claim that  $\beta$  is equivalent to the following formula

$$\bigvee_{i \in [k]} G_{g_i}^F \langle \phi_1, \dots, \phi_k \rangle \wedge P H G_{g.g_i}^F \langle \phi_1, \dots, \phi_k \rangle$$

The formula says that, if the “result of the group computation” in the future of the current position is  $g_i$ , then the value of the group computation from the beginning of the word should be  $g.g_i$ . See Figure 1.  $\square$

Observe that the formula  $\hat{\alpha}$  in the proof above is an impure formula, even if  $\alpha$  was one. This takes us to the next section, where we show that any

formula in LTLGRP (LTLMOD) can be written as a boolean combination of pure formulas.

### 3.1 Separation property of LTLGRP

We say that a logic satisfies the *separation* property [Gab87] if any formula in that logic can be written as a boolean combination of pure past, pure present and pure future formulas.

**Theorem 3.3** [Gab87] *The logic LTL satisfies the separation property*

We next show that logics LTLGRP, LTLMOD have the separation property. The proof is given by a series of technical lemmas. The translations given below in Lemma 3.4 is the base case for the full proof.

**Lemma 3.4** *The following formulas can be separated.*

- (i)  $\alpha = G_g^F \langle (aSb \wedge \beta) \vee \phi_1, \dots, \phi_k \rangle$
- (ii)  $\alpha = G_g^F \langle (G_{g'}^P \langle \gamma_1, \dots, \gamma_k \rangle \wedge \beta) \vee \phi_1, \dots, \phi_k \rangle$
- (iii)  $\alpha_1 S(G_{g'}^P \langle \gamma_1, \dots, \gamma_k \rangle \wedge \alpha_2)$
- (iv)  $(G_{g'}^P \langle \gamma_1, \dots, \gamma_k \rangle \vee \alpha_2) S\alpha_1$
- (v)  $\alpha = G_g^F \langle (aUb \wedge \beta) \vee \phi_1, \dots, \phi_k \rangle$
- (vi)  $\alpha = G_g^F \langle (G_{g'}^F \langle \gamma_1, \dots, \gamma_k \rangle \wedge \beta) \vee \phi_1, \dots, \phi_k \rangle$
- (vii)  $\alpha_1 U(G_{g'}^F \langle \gamma_1, \dots, \gamma_k \rangle \wedge \alpha_2)$
- (viii)  $(G_{g'}^F \langle \gamma_1, \dots, \gamma_k \rangle \vee \alpha_2) U\alpha_1$

**Proof.**

**(1):** Let  $x$  be the current location and  $z > y > x$  be locations such that  $b$  is true at  $y$  and  $a$  is true at all positions in the interval  $(y, z)$  and  $z$  satisfies  $\beta$ . That is  $z$  is where  $aSb \wedge \beta$  is true. See Figure 2. Observe that  $(y, z)$  is a block of states which satisfy  $aSb$  formula. Our idea is to get the group value computed in this interval. We consider the case where  $x$  does not satisfy the formula  $aSb$ . The solution we give can be modified to take this also into consideration.

Let  $\Theta = \langle \phi_1, \dots, \phi_k \rangle$ . We now give a formula  $\psi_g$  which is true at all positions which satisfy  $b \wedge aU\beta$  and the group value computed for the block of  $as$  until  $\beta$  is  $g$ .

$$\psi_g := \bigvee_{g'=g.g''} b \wedge (aU(\beta \wedge G_{g''}^F \Theta)) \wedge G_g^F \Theta$$

Thus  $\psi_g$  is true at  $y$  iff the group value in the interval  $(y, z)$  is  $g$ . Let  $\gamma = \neg(aU\beta)$ . Now the following formula is equivalent to  $\alpha$ .

$$G_g^F \langle (\gamma \wedge \phi_1) \vee \psi_{g_1}, \dots, (\gamma \wedge \phi_k) \vee \psi_{g_k} \rangle$$

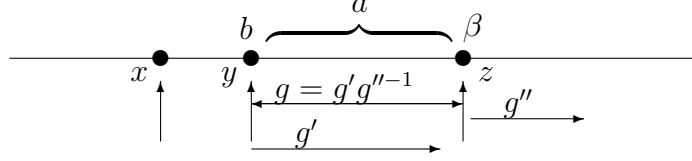


Fig. 2. Timeline for  $aSb \wedge \beta$

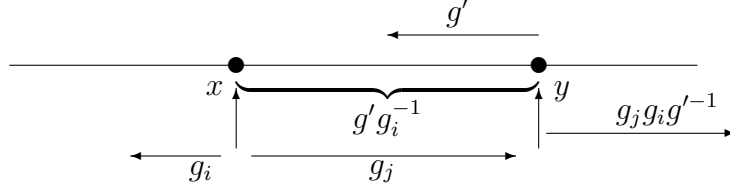


Fig. 3. Timeline for  $G_{g'}^P(\dots)$

The formula evaluates  $\phi_i$ s only when the position does not satisfy  $aU\beta$ . Otherwise one of the  $\psi_{g_i}$ s would have calculated the group value for the entire block.

**(2,3,4):** Let  $\Delta(i, j, \langle \gamma_1, \dots, \gamma_k \rangle)$  be true at the current position. Then any point in the future will satisfy  $G_{g'}^P\langle \gamma_1, \dots, \gamma_k \rangle$  iff it also satisfies  $G_{g_jg_ig'^{-1}}^F\langle \gamma_1, \dots, \gamma_k \rangle$ . This lets us replace the past group operator with a future group operator and vice versa. See Figure 3.

**(5,6,7,8):** These formulas are got by replacing past operators with future operators and vice versa in the formulas in 1,2,3,4. By the same arguments above we can show that these formulas can also be separated.  $\square$

**Lemma 3.5** *The following translations are equivalent.*

- (i)  $\alpha_1 S(\alpha_2 \vee \alpha_3) = \alpha_1 S\alpha_2 \vee \alpha_1 S\alpha_3$
- (ii)  $(\alpha_1 \wedge \alpha_2) S\alpha_3 = \alpha_1 S\alpha_3 \wedge \alpha_2 S\alpha_3$
- (iii)  $\neg(aSb) = (\neg bS\neg a) \vee H\neg b$
- (iv)  $\neg G_g^P\langle \phi_1, \dots, \phi_k \rangle = \bigvee_{g \neq g_i} G_{g_i}^P\langle \phi_1, \dots, \phi_k \rangle$
- (v)  $\alpha_1 U(\alpha_2 \vee \alpha_3) = \alpha_1 U\alpha_2 \vee \alpha_1 U\alpha_3$
- (vi)  $(\alpha_1 \wedge \alpha_2) U\alpha_3 = \alpha_1 U\alpha_3 \wedge \alpha_2 U\alpha_3$
- (vii)  $\neg(aUb) = (\neg bU\neg a) \vee G\neg b$
- (viii)  $\neg G_g^F\langle \phi_1, \dots, \phi_k \rangle = \bigvee_{g \neq g_i} G_{g_i}^F\langle \phi_1, \dots, \phi_k \rangle$

**Proof.** It is easy to see that the fourth and eighth statements are correct. The rest of the statements can be proved as shown in [Gab87].  $\square$

Lemma 3.4 and Lemma 3.5 lets us rewrite formulas with past operators nested inside future operators. Observe that there are dual Lemmas where the past operator is replaced by the future operator and vice versa. Using these two lemmas we give a series of lemmas to show that formulas in LTLGRP can



be separated. These lemmas are proved using induction on the structure of the formula.

**Lemma 3.6** (i) *Let  $a, b$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having the only modality being  $aUb$ . Then  $\alpha S\beta, G_g^P\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

(ii) *Let  $a, b$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having the only modality being  $aSb$ . Then  $\alpha U\beta, G_g^F\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

**Proof.** Observe that the (ii)nd statement is the (i)st statement with past modalities replaced by future modalities and vice versa. We prove (i) now. Gabbay [Gab87] shows how to separate the formula  $\alpha S\beta$ . So let us consider the formula  $G_g^P\langle\phi_1, \dots, \phi_k\rangle$ . Let each  $\phi_i$  be a boolean combination of  $aUb$  and propositions. To rewrite  $G_g^P\langle\phi_1, \dots, \phi_k\rangle$  as boolean combination of pure Future and pure Past formulas, we apply the transformations given in Lemma 3.4 repeatedly which gives us a separated formula.  $\square$

**Lemma 3.7** (i) *Let  $a_1, \dots, a_k$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality  $G_g^F\langle a_1, \dots, a_k\rangle$ . Then  $\alpha S\beta, G_{g'}^P\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

(ii) *Let  $a_1, \dots, a_k$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality  $G_g^P\langle a_1, \dots, a_k\rangle$ . Then  $\alpha U\beta, G_{g'}^F\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

**Proof.** Repeated application of Lemma 3.4 and Lemma 3.5 give us a separated formula.  $\square$

We now look at formulas where an until modality (since modality) is nested inside a past modality (future modality).

**Lemma 3.8** (i) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality being  $\forall i \leq n : U_i = a_i U b_i$ . Then  $\alpha S\beta, G_g^P\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

(ii) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality being  $\forall i \leq n : S_i = a_i S b_i$ . Then  $\alpha U\beta, G_g^F\langle\phi_1, \dots, \phi_k\rangle$  can be separated.*

**Proof.** We prove (i) and claim that the proof for (ii) is similar. When  $\psi = \alpha S\beta$ , this can be separated by the arguments of Gabbay. So let  $\psi = G_g^P\langle\phi_1, \dots, \phi_k\rangle$ . Let  $\{U_1, \dots, U_n\}$  be the  $n$  Until formulas used in the  $\phi_i$ s. We first replace the Until formulas  $U_1, \dots, U_{n-1}$  by new propositions  $p_1, \dots, p_{n-1}$ . Let the new formula be called  $\hat{\psi}$ . By Lemma 3.6 we know that we can find a separated formula equivalent to  $\hat{\psi}$ . Now replace  $p_{n-1}$  in the formula by  $U_{n-1}$  and again apply Lemma 3.6. Observe that we did not introduce any

new Untils when we separated. After  $n$  rounds we get a formula which is separated.  $\square$

**Lemma 3.9** (i) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality  $G_g^F \langle a_1, \dots, a_k \rangle$ . Then  $\alpha S\beta, G_{g'}^P \langle \phi_1, \dots, \phi_k \rangle$  can be separated.*

(ii) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  and  $\phi_1, \dots, \phi_k$  be formulas having only the modality  $G_g^P \langle a_1, \dots, a_k \rangle$ . Then  $\alpha U\beta, G_{g'}^F \langle \phi_1, \dots, \phi_k \rangle$  can be separated.*

**Proof.** The proof is similar to the proof of Lemma 3.8. In (i) we replace  $\forall i < n, G_g^F \langle a_1, \dots, a_k \rangle$  by new propositions  $p_i$ . We then apply 3.6 and continue as in the proof of Lemma 3.8.  $\square$

**Lemma 3.10** (i) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  be formulas having only the modality  $\forall i \leq n : U_i = a_i U b_i s$  or  $G_g^F \langle a_1, \dots, a_k \rangle$ . Then  $\alpha S\beta, G_{g'}^P \langle \phi_1, \dots, \phi_k \rangle$  can be separated.*

(ii) *Let  $a_1, \dots, a_n, b_1, \dots, b_n$  be boolean combinations of propositional formulas. Let  $\alpha, \beta$  be formulas having only the modality  $\forall i \leq n : U_i = a_i S b_i s$  or  $G_g^P \langle a_1, \dots, a_k \rangle$ . Then  $\alpha U\beta, G_{g'}^F \langle \phi_1, \dots, \phi_k \rangle$  can be separated.*

**Proof.** The proof is by combining the two Lemma 3.8 and Lemma 3.9.  $\square$

Now we look at formulas having Until (Since) modality but without any Since (Until) nested inside an Until or Since. That is no modality is nested inside a Future (Past) modality, but Past and Future modalities can be nested with Past (Future) modality.

**Lemma 3.11** (i) *Let  $a_1, \dots, a_k, b_1, \dots, b_k$  be boolean combinations of propositional formulas. Let  $\alpha$  be a formula such that the Future modalities are of the form  $\forall i \leq n : U_i = a_i U b_i s$  or  $G_g^F \langle a_1, \dots, a_k \rangle$ . Then  $\alpha$  can be separated.*

(ii) *Let  $a_1, \dots, a_k, b_1, \dots, b_k$  be boolean combinations of propositional formulas. Let  $\alpha$  be a formula such that the Future modalities are of the form  $\forall i \leq n : U_i = a_i S b_i s$  or  $G_g^P \langle a_1, \dots, a_k \rangle$ . Then  $\alpha$  can be separated.*

**Proof.** We prove (i) and claim that the proof for (ii) is similar. Let the Past depth be  $n$ . If  $n = 0$  the claim is trivially true. When  $n = 1$  the claim follows from Lemma 3.10. For depth  $n > 1$  we repeatedly apply Lemma 3.10 to the most deeply nested Past modality. After each application of the Lemma the depth of the Past modality is reduced and hence after  $n$  steps we get a separated formula.  $\square$

Now we consider formulas which can have Future modalities nested inside the Past modality.

**Lemma 3.12** *Let  $\alpha$  be a formula such that no Past (Future) modality is nested inside a Future (Past) modality. Then  $\alpha$  can be separated.*

**Proof.** The proof is by induction on the depth  $n$  of the Future (Past) modality.  $n = 1$  was proved by Lemma 3.11. When  $n > 1$ , we replace all Future (Past) modalities at Future (Past) depth  $\geq 2$  by new propositions  $p_i$ . Let the resultant formula be  $\hat{\alpha}$ . Observe that the Future (Past) depth of  $\hat{\alpha}$  is one and hence can be separated by Lemma 3.11. Now replace all the  $p_i$ s by the Future (Past) modalities we replaced them with. Observe that we have reduced the Future (Past) depth. We repeat the above process until we get a separated formula.  $\square$

Finally we show that any formula  $\alpha \in \text{LTLGRP}$  can be separated.

**Theorem 3.13** *Let  $\alpha$  be an LTLGRP (LTLMOD) formula. Then  $\alpha$  can be separated.*

**Proof.** This involves induction on the alternation depth  $n$  of the formula  $\alpha$ .  $n = 1$  was proved in Lemma 3.12. When  $n > 1$ , we replace the modalities by propositions such that we get a formula  $\hat{\alpha}$  which is of alternation depth one. Lemma 3.12 will give a separated  $\hat{\alpha}'$  formula equivalent to  $\hat{\alpha}$ . Now replace the propositions in  $\hat{\alpha}'$  with the modalities we had earlier replaced with. The formula we get is of alternation depth lesser than  $\alpha$ . Hence we can repeat the procedure until we get a formula which is separated.  $\square$

**Corollary 3.14** *Every LTLGRP (LTLMOD) formula is initially equivalent to a formula with only future modalities.*

**Proof.** Let  $\alpha$  be an LTLGRP (LTLMOD) formula. By Theorem 3.13  $\alpha$  can be separated. We can now replace the past formulas with *false* since all statements regarding past are false at the  $0^{\text{th}}$  position. The resultant formula which is equivalent to  $\alpha$  now contains only future modalities.  $\square$

### 3.2 Expressive Completeness of LTLGRP

**Lemma 3.15** *LTLGRP (LTLMOD) has separation property iff it is expressively complete for FOGRP (FOMOD).*

**Proof.** ( $\Leftarrow$ ): Let  $\alpha$  be an LTLGRP (LTLMOD) formula. We can now write a first order logic formula on free variable  $x$ ,  $\alpha'(x)$  such that it is equivalent to  $\alpha$ . First order logic formulas can be separated using relativization. This can be proved by induction on the structure of the formula. The atomic case is trivial. Formulas of form  $\exists y\phi(x, y)$  can be replaced by  $\exists y < x\phi(x, y) \vee \phi(x, x) \vee \exists y > x\phi(x, y)$ . A similar proof can be given for group quantifiers too.

Now since LTLGRP (LTLMOD) is expressively complete for FOGRP (FOMOD)

each of the seperated formulas can be replaced with LTLGRP (LTLMOD) formulas. This gives us a seperated formula.

( $\Rightarrow$ ): We show that for an FOGRP (FOMOD) formula with one free variable we can give an equivalent LTLGRP (LTLMOD) formula. Let  $P_1, \dots, P_n$  be the unary predicates. The proof is by induction on the quantifier depth. For the base case we assume formulas with no quantifiers. This consists of boolean combination of formulas of the form  $P_i(x)$ . The translation of this formula will be boolean combination of formulas of the form  $p_i$ .

Now let us assume that all FOGRP formulas with one free variable and of quantifier depth  $< k$  over any constant number of unary predicates (alphabet) can be converted into an LTLGRP formula. Let  $Q$  be a quantifier. Consider the formula  $\psi(x) = Qy \phi(x, y)$  such that  $\phi$  is of quantifier depth  $< k$ . We first remove  $x$  from  $\psi$  as follows. All subformulae of the form  $x = x, x < x, x > x$  are replaced by  $\top, \perp, \perp$  respectively. Now we rewrite  $\psi$  as follows

$$\psi(x) = \bigvee_{\bar{v}=\{0,1\}^n} \left( \left( \bigwedge_{i=1}^n P_i(x) \Leftrightarrow v_i \right) \Rightarrow \psi^v(x) \right)$$

Here  $\psi^v(x)$  replaces all occurrences of subformulas of the form  $P_i(x)$  with  $\top, \perp$  depending on  $v_i$ . Now the subformulas in each of  $\psi^v$  containing  $x$  will be of the form  $x < z, x > z, x = z$ , where  $z$  is some other variable in  $\psi$ . We remove these formulae by introducing three new unary predicates  $R_<, R_>, R_=$  and replacing  $x \text{ op } z$  by  $R_{op}(z)$  ( $\text{op} := \{<, >, =\}$ ). The resultant formula  $\psi^v$  will not contain any occurrence of  $x$ . Moreover if we assume the interpretations for  $R_{op}$  it will be equivalent to the old formula. Now let  $\psi^v = Qy\alpha^v(y)$ .

**Case 1,  $Q = \exists$ :** Since  $\alpha^v(y)$  is a formula with one free variable and quantifier depth  $< k$ , we can apply the inductive hypothesis to get an LTLGRP formula  $\gamma$  with new propositions  $r_<, r_>, r_=$ . We now write  $\beta = \text{P}\gamma \vee \gamma \vee \text{F}\gamma$ . Since LTLGRP formulas can be seperated we can now seperate  $\beta$  into boolean combinations of pure past, pure present and pure future formulas. Finally in all the pure past formulas we replace  $r_<, r_>, r_=$  with  $\top, \perp, \perp$  respectively. Similarly one can replace all the  $r_{op}$  formulae with  $\top, \perp$  in the pure future, and pure present formulae.

**Case 2,  $Q = G_g$ :** So let  $\psi^v = G_g y \langle \alpha_1(y), \dots, \alpha_k(y) \rangle$ . Since  $\alpha_i^v(y)$ s are formulas with one free variable and quantifier depth  $< k$ , we can apply the inductive hypothesis to get LTLGRP formulas  $\phi_i$ s with new propositions  $r_<, r_>, r_=$ .

Let us denote by  $\Phi = \langle \phi_1, \dots, \phi_k \rangle$ . Then we can write  $\beta = \bigvee_{i,j,l} G_{g_i}^P \Phi \wedge \Gamma_l(\Phi) \wedge G_{g_j}^F \Phi$ , such that  $g_i \cdot g_l \cdot g_j = g$  and  $\psi^v = \beta$ . Here  $\Gamma_l(\Phi)$  is true only if  $(\bigwedge_{j=1}^l \neg \phi_j) \wedge \phi_l$

We can now separate  $\beta$  into boolean combination of pure past, pure present and pure future formulas. Finally in all the pure past formulas we replace  $r_<, r_>, r_=$  with  $\top, \perp, \perp$  respectively. Similarly one can replace all the  $r_{op}$  for-

formulae with  $\top, \perp$  in the pure future, and pure present formulae. So we have shown that the formula  $\psi(x)$  has an equivalent LTLGRP formula. Replacing the  $P_i(x)$  with  $p_i$  in the rest of the  $\psi$  formula will give us an LTLGRP formula which is equivalent to  $\psi$ .  $\square$

Lemma 3.15 along with Theorem 3.13 gives us that

**Theorem 3.16** *LTLGRP (LTLMOD) is expressively complete for FOGRP (FOMOD).*

As a corollary we get

**Corollary 3.17** *Every FOGRP (FOMOD) formula with one free variable has an equivalent formula using three variables.*

**Proof.** Let  $\phi(x)$  be a FOGRP (FOMOD) formula. By Theorem 3.16 we know that there exists an equivalent LTLGRP (LTLMOD) formula  $\psi$ . We now inductively built a FOGRP (FOMOD) formula from  $\psi$  as follows. The translation,  $t$  is inductively given as follows.

LTLGRP	FOGRP
$p$	$P(x)$
$\alpha \vee \beta$	$t(\alpha) \vee t(\beta(x))$
$\neg\alpha$	$\neg t(\alpha(x))$
$\alpha \mathbf{U} \beta$	$\exists y (y > x) \wedge t(\beta(y)) \wedge \forall z (x < z < y) \implies t(\alpha(z))$
$\alpha \mathbf{S} \beta$	$\exists y (y < x) \wedge t(\beta(y)) \wedge \forall z (y < z < x) \implies t(\alpha(z))$
$G_g^F \langle \phi_1, \dots, \phi_k \rangle$	$G_g y \langle (x < y) \wedge \phi_1(y), \dots, (x < y) \wedge \phi_k(y) \rangle$
$G_g^P \langle \phi_1, \dots, \phi_k \rangle$	$G_g y \langle (x > y) \wedge \phi_1(y), \dots, (x > y) \wedge \phi_k(y) \rangle$

Clearly this translation uses only three variables and hence we get an equivalent FOGRP (FOMOD) formula in three variables.  $\square$

## 4 UTLgrp and two variable fragment of FOgrp

Here we show that UTLGRP is expressively complete for the two variable logic fragment of FOGRP, (written as  $\text{FO}^2\text{GRP}$ ).

**Theorem 4.1** *UTLGRP is expressively complete for  $\text{FO}^2\text{GRP}$ .*

**Proof.** The translation is recursive. For the atomic formulas, boolean combinations and existential formulas we follow the proof by Etessami et al [EVW02]. We give here the translation for the group quantifier (A similar translation can be given for the modulo quantifiers). Let

$$\phi(x) := G_g y \langle \phi^1(x, y), \dots, \phi^k(x, y) \rangle$$

Each of the  $\phi^j(x, y)$ , for  $j \leq k$  can be rewritten as

$$\phi^j(x, y) := \tau^j(\gamma_1^j(x, y), \dots, \gamma_r^j(x, y), \alpha_1^j(x), \dots, \alpha_s^j(x), \beta_1^j(y), \dots, \beta_t^j(y))$$

Here  $\tau^j$ s are boolean propositional formula.  $\gamma_i^j$ s are order formulas of the form  $x < y, x = y, x > y$ . The successor relations inside the group quantifiers can be removed and hence we dont consider it here.  $\alpha_i^j$ s and  $\beta_i^j$ s are formulas whose quantifier depth is less than the quantifier depth of  $\phi(x)$ . Moreover  $\alpha_i^j$ s and  $\beta_i^j$ s are of type atomic or existential or group quantified. We first take out the  $\alpha_i^j$ s outside the group quantifier. For a vector  $\mathbf{v} = (v_1, \dots, v_s) \in \{0, 1\}^s$  we define

$$\psi_{\mathbf{v}}^j(x, y) = \tau^j(\gamma_1^j(x, y), \dots, \gamma_r^j(x, y), v_1, \dots, v_s, \beta_1^j(y), \dots, \beta_t^j(y))$$

Let us denote by  $\psi^g$  for a  $g \in \langle G \rangle$  (an element generated by  $G$ ) the formula:  $\psi^g(x) := G_g y \langle \psi_{\mathbf{v}}^1(x, y), \dots, \psi_{\mathbf{v}}^k(x, y) \rangle$ . Thus we can rewrite  $\phi$  as follows.

$$\phi(x) := \bigvee_{\mathbf{v} \in \{T, F\}^s} \left( \bigwedge_{i < s} \alpha_i^j(x) \leftrightarrow v_i \right) \wedge \psi^g(x)$$

Observe that the  $\gamma_i^j$ s are order formulas. Let  $\Gamma = \{x < y, x = y, x > y\}$ , be the set of all order relations between  $x$  and  $y$ . For any order relation,  $o \in \Gamma$ ,  $\gamma_i^j$ s will be evaluated to  $\{T, F\}$ . Let  $\psi^o$  be the formula got by replacing  $\gamma_i^j$  with  $T/F$  in  $\psi(x)$  depending on the order  $o$ . Observe that  $x$  doesnt appear free in  $\psi^o$ . Thus we get

$$\phi(x, y) := \bigvee_{\mathbf{v} \in \{T, F\}^s} \left( \bigwedge_{i < s} \alpha_i(x) \leftrightarrow v_i \right) \wedge \bigvee_{g_1 g_2 g_3 = g} \phi_{g_1}^{y < x} \wedge \phi_{g_2}^{x = y} \wedge \phi_{g_3}^{y > x}$$

Since the formulas  $\psi^o$  does not contain any free variable we can recursively build the UTLGRP formula. The  $\tau_{\mathbf{v}}^j(y)^o$  formulas have UTLGRP equivalent formulas, since the quantifier depth in  $\tau_{\mathbf{v}}^j(y)^o$  is less than  $\phi(x)$ .

□

## References

- [BB87] Behnam Banieqbal and Howard Barringer. Temporal logic with fixed points. In *Temporal Logic in Specification*, pages 62–74, London, UK, 1987. Springer-Verlag.

- [BIS90] David A. Barrington, Neil Immerman, and Howard Straubing. On uniformity within  $NC^1$ . *Journal of Computer and System Sciences*, 41(3):274–306, December 1990.
- [BMT99] Augustin Baziramwabo, Pierre McKenzie, and Denis Thérien. Modular temporal logic. In *Proc. 14th LICS*, page 344. IEEE, 1999.
- [EVW02] Kousha Etessami, Moshe Y. Vardi, and Thomas Wilke. First-order logic with two variables and unary temporal logic. *Inf. Comput.*, 179(2):279–295, 2002.
- [Gab87] Dov M. Gabbay. The declarative past and imperative future: Executable temporal logic for interactive systems. In *Temporal Logic in Specification*, pages 409–448, London, UK, 1987. Springer-Verlag.
- [GPSS80] Dov Gabbay, A. Pnueli, S. Shelah, and J. Stain. On the temporal logic of programs. In *Symposium on mathematical foundations of CS 18*, volume 88 of *LNCS*, pages 112–152, 1980.
- [Hod95] Ian M. Hodkinson. On gabbay’s temporal fixed point operator. *Theor. Comput. Sci.*, 139(1&2):1–25, 1995.
- [HT99] Jesper G. Henriksen and P.S. Thiagarajan. Dynamic linear time temporal logic. *Ann. Pure Appl. Logic*, 96(1-3):187–207, 1999.
- [Hus01] Ullrich Hustadt. Temporal logic: Mathematical foundations and computational aspects, volume 2, dov m. gabbay, mark a. reynolds, and marcelo finger. *J. of Logic, Lang. and Inf.*, 10:406–410, June 2001.
- [JW96] David Janin and Igor Walukiewicz. On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic. In *Proceedings of the 7th International Conference on Concurrency Theory, CONCUR ’96*, pages 263–277, London, UK, 1996. Springer-Verlag.
- [Kam68] J. A. W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, CA, 1968.
- [LS10] Kamal Lodaya and A.V. Sreejith. Ltl can be more succinct. In *Proc. 8th ATVA, Singapore*, volume 6252 of *LNCS*, pages 245–258, 2010.
- [Pnu77] A. Pnueli. The temporal logic of programs. In *focs77*, pages 46–57, 1977.
- [Pri56] Arthur N. Prior. *Time and Modality*. Oxford University Press, Oxford, 1956.
- [RS71] R. McNaughton and S. Papert. *Counter-free Automata*. MIT Press, Cambridge, USA, 1971.
- [SM73] L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the fifth annual ACM symposium on Theory of computing, STOC ’73*, pages 1–9, New York, NY, USA, 1973. ACM.
- [STT95] Howard Straubing, Denis Thérien, and Wolfgang Thomas. Regular languages defined with generalized quantifiers. *Inf. Comput.*, 118(3):389–301, 1995.
- [Wol83] Pierre Wolper. Temporal logic can be more expressive. *Inf. Contr.*, 56(1-2):72–99, 1983.