Available online at www.sciencedirect.com

SciVerse ScienceDirect

Electronic Notes in Theoretical Computer Science 278 (2011) 201–214

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Expressive Completeness for LTL With Modulo Counting and Group Quantifiers

A V Sreejith [1]

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 [14], Pnueli [13], Gabbay, Pnueli, Shelah and Stavi [5]) is expressively complete for first order logic FO with monadic predicates. Kamp's proof ([11]) also establishes that FO has the three-variable property: every first order formula has an equivalent formula using just three variables. Gabbay [6] 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 [15] showed that these logics also correspond to the class of starfree regular languages. The translation algorithms are not elementary, and from Meyer and Stockmeyer [16] we know we cannot do better.

Various extensions have been proposed to extend the expressiveness of temporal logic over word models to all the regular languages (Wolper [18], Banieqbal and Barringer [1], Hodkinson [9], Baziramwabo and McKenzie and Thérien [3], Henriksen

[1] Email: sreejith@imsc.res.in

doi:10.1016/j.entcs.2011.10.016

and Thiagarajan [8]). Hodkinson's proof (see the book ([7])) represents finite automata using LTL with fixpoint operators, nowadays called the linear mu-calculus. The later work of Janin and Walukiewicz ([10]), which is in a branching context, also passes through automata.

Barrington, Immerman and Straubing [2] 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 ([17]). Baziramwabo, McKenzie, Thérien [3] 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 the logic LTLMOD is expressively complete for FOMOD.

In this article we give Gabbay-style [6] 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 [3] and in our own earlier work ([12]).

We also look at unary temporal logic UTL ([4]), which is LTL without the Until operator. We extend the technique of Etessami, Vardi and Wilke ([4]) to show that UTL with group operators (UTLGRP) has the same expressive power as FOGRP which uses only two variables.

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 [2], 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 gives 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

$$\phi ::= p \in \mathcal{P} \mid \neg\, \phi \mid \phi_1 \;\vee\; \phi_2 \mid \mathsf{X}\phi \mid \mathsf{Y}\phi \mid \mathsf{F}\ \phi \mid \mathsf{P}\ \phi \mid \phi_1\ \mathsf{U}\ \phi_2 \mid \phi_1\ \mathsf{S}\ \phi_2$$

We denote by $true$ the statement $a \vee \neg a$. Then $\mathsf{F}\phi$, $\mathsf{P}\phi$ can be defined as $true\mathsf{U}\phi$ and $true\mathsf{S}\phi$ 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^{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 ϕ

over $\mathcal{P}$, we denote by $(u,i) \models \phi$ to mean that ϕ is true at position i in the word u. The *semantics* of the logic is given below

$$\begin{array}{lcl}
(u,i) & \models & p \text{ if } p \in u(i) \text{ and } p \in \mathcal{P} \\
(u,i) & \models & \neg\phi \text{ if not } (u,i) \models \phi \\
(u,i) & \models & \phi_1 \vee \phi_2 \text{ if } (u,i) \models \phi_1 \text{ or } (u,i) \models \phi_2 \\
(u,i) & \models & \mathsf{X}\phi \text{ if } i < |u| - 1 \text{ and } (u,i+1) \models \phi \\
(u,i) & \models & \mathsf{Y}\phi \text{ if } i > 0 \text{ and } (u,i-1) \models \phi \\
(u,i) & \models & \phi_1\mathsf{U}\phi_2 \text{ if there exists a } j > i \text{ and } (u,j) \models \phi_2 \text{ and for all } k, \\
 & & \text{ if } i < k < j \text{ then } (u,k) \models \phi_1 \\
(u,i) & \models & \phi_1\mathsf{S}\phi_2 \text{ if there exists a } j < i \text{ and } (u,j) \models \phi_2 \text{ and for all } k, \\
 & & \text{ if } j < k < i \text{ then } (u,k) \models \phi_1
\end{array}$$

As usual $\mathsf{G}\phi$ abbreviates $\neg\mathsf{F}\neg\phi$ and $\mathsf{H}\phi$ abbreviates $\neg\mathsf{P}\neg\phi$. Restricting the above logic to not have since (S) and until (U) operator, gives us the logic UTL.

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

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 $r \leq q$:

$$MOD^{F}_{r,q}\phi \mid MOD^{P}_{r,q}\phi$$

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

$$G^{F}_{g,(g_1,\ldots,g_k)}\langle\phi_1,\ldots,\phi_k\rangle \mid G^{P}_{g,(g_1,\ldots,g_k)}\langle\phi_1,\ldots,\phi_k\rangle$$

Here G is a finite group whose elements are $\{g_1,\ldots,g_k,1\}$ and g is an element in G. Here 1 denotes the identity of 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 ϕ 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.

$$u,i \models MOD^{F}_{r,q}\phi \text{ iff } |\{i < l < |u| \mid u,l \models \phi\}| \equiv_q r$$

$$u,i \models MOD^{P}_{r,q}\phi \text{ iff } |\{0 \leq l < i \mid u,l \models \phi\}| \equiv_q r$$

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

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

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

Observe that this is a generalization of the modulo counting we did earlier, since modulo counting is similar to working with cyclic groups. For example. $MOD_{1,q}^F\phi$ can be expressed by the following formula which uses the cyclic group, $G = \{g_1, g_1^2, g_1^3, \ldots, g_1^{|q|}\}$.

$$G_{g_1}^F\langle \phi, false, \ldots, false\rangle$$

Later we use the notation $\Gamma_j\langle \phi_1, ..., \phi_k\rangle$ to denote the formula $\neg\phi_1 \wedge \cdots \wedge \neg\phi_{j-1} \wedge \phi_j$

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 $\mathsf{F}\ MOD_{r,q}^P\alpha$ 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 $\phi_1\mathsf{U}\phi_2$, $\mathsf{X}\phi_1$, $G_g^F\langle \phi_1, ..., \phi_k\rangle$ is one more than the maximum of the future depth of the formulas $\phi_1, ..., \phi_k$. Similarly the past depth of the formulas $\phi_1\mathsf{S}\phi_2$, $\mathsf{Y}\phi_1$, $G_g^P\langle \phi_1, ..., \phi_k\rangle$ is one more than the past depth of the formulas $\phi_1, ..., \phi_k$.
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 Σ be a finite alphabet and $\mathcal{V}$ be a finite set of variables. A *word model* over $(\Sigma, \mathcal{V})$ is a pair (u, s), where $u \in \Sigma^*$ and $s : \mathcal{V} \to \{1, ..., m\}$, where $|u| = m$. Let us introduce the syntax of first order logic for word models. We use a finite set of variables $\mathcal{V} = \{x_1, ...\}$, a binary predicate $<$ and unary predicates a, $a \in \Sigma$, for the finite alphabet Σ.

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

The semantics for first order logic is standard. In the logic FOmod, introduced by Straubing, Thérien and Thomas [17], one can count the number of times a formula is satisfied modulo a number:

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

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

$$(u,s) \models \exists^{r,q} x\phi \text{ iff } |\{l \in \{1,...,|u|\} | (u, s[x \mapsto l]) \models \phi\}| \equiv (r \mod q)$$

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 [2]:

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

Its semantics is given as follows.

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

The definition of Γ is the same as the one given in LTLgrp semantics.

The following theorem [11,6] identifies the set of languages accepted by LTL.

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

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

$$u, i \vDash \phi \Leftrightarrow u, i \vDash \psi$$

We show in the next section that the theorem continous 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 $\alpha \in$ LTLgrp. Then there exists a group, G_α and a formula $\hat{\alpha} \in$ LTLgrp(G_α) such that $L(\alpha) = L(\hat{\alpha})$.*

Proof. Take G_α to be the cross product of all the groups in α. Now any group G can be replaced by G_α (also requires modifying accepting group element). □

Hence we can always work with formulas over 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 $\alpha \in$ LTLgrp. Then α is equivalent to a formula $\hat{\alpha}$, where $\hat{\alpha}$ do not contain any past group operator.*

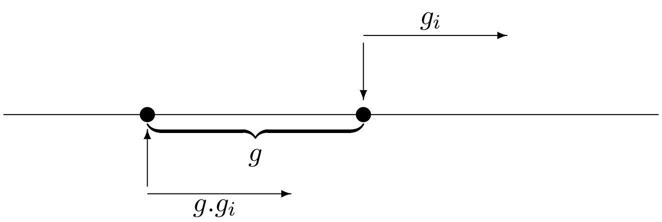

Fig. 1. Future Group modality can simulate Past Group modality

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

$$\bigvee_{i\in[k]} G^F_{g_i}\langle\phi_1, ..., \phi_k\rangle \wedge \mathsf{P}\ \mathsf{H}\ G^F_{g.g_i}\langle\phi_1, ..., \phi_k\rangle$$

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 $g.g_i$. Note that both the above formula and β are not satisfied at the beginning of the word since we use strict past. See Figure 1 for a position not at the beginning of the word. □

Observe that the formula $\hat{\alpha}$ in the proof above is an impure formula, even if α 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 formula can be *separated* if it can be written as a boolean combination of pure past, pure present and pure future formulas. We say that a logic satisfies the *separation* property [6] if all formulas in that logic can be separated.

Theorem 3.3 *[6] 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) $G^F_g\langle(a\mathit{S}b \wedge \beta) \vee \phi_1, ..., \phi_k\rangle$

(ii) $G^F_g\langle(G^P_{g'}\langle\gamma_1, ..., \gamma_k\rangle \wedge \beta) \vee \phi_1, ..., \phi_k\rangle$

(iii) $\alpha_1\mathit{U}(G^P_{g'}\langle\gamma_1, ..., \gamma_k\rangle \wedge \alpha_2)$

(iv) $(G^P_{g'}\langle\gamma_1, ..., \gamma_k\rangle \vee \alpha_2)\mathit{U}\alpha_1$

(v) $G^P_g\langle(a\mathit{U}b \wedge \beta) \vee \phi_1, ..., \phi_k\rangle$

(vi) $G^P_g\langle(G^F_{g'}\langle\gamma_1, ..., \gamma_k\rangle \wedge \beta) \vee \phi_1, ..., \phi_k\rangle$

(vii) $\alpha_1\mathit{S}(G^F_{g'}\langle\gamma_1, ..., \gamma_k\rangle \wedge \alpha_2)$

(viii) $(G^F_{g'}\langle\gamma_1, ..., \gamma_k\rangle \vee \alpha_2)\mathit{S}\alpha_1$

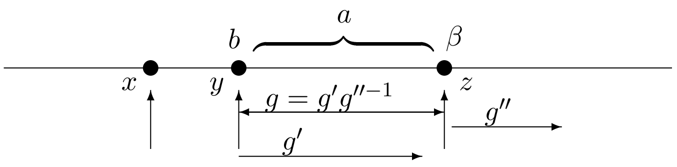

Fig. 2. Timeline for $a\mathsf{S}b \wedge \beta$

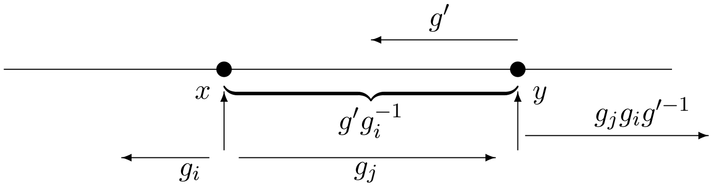

Fig. 3. Timeline for $G^P_{g'}\langle ...\rangle$

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 β. That is z is where $a\mathsf{S}b \wedge \beta$ is true. See Figure 2. Observe that (y, z) is a block of states which satisfy $a\mathsf{S}b$ formula. Our idea is to get the group value computed in this interval. We consider the case where x does not satisfy the formula $a\mathsf{S}b$. The solution we give can be modified to take this also into consideration.
Let $\Theta = \langle \phi_1, ..., \phi_k\rangle$. We now give a formula ψ_g which is true at all positions which satisfy $b \wedge a\mathsf{U}\beta$ and the group value computed for the block of as until β is g.

$$\psi_g := \bigvee_{g'=g.g''} b \wedge (a\mathsf{U}(\beta \wedge G^F_{g''}\Theta)) \wedge G^F_{g'}\Theta$$

Thus ψ_g is true at y iff the group value in the interval (y, z) is g. Let $\gamma = \neg(a\mathsf{U}\beta)$. Now the following formula is equivalent to α.

$$G^F_g\langle(\gamma \wedge \phi_1) \vee \psi_{g_1}, ..., (\gamma \wedge \phi_k) \vee \psi_{g_k}\rangle$$

The formula evaluates ϕ_is only when the position does not satisfy $a\mathsf{U}\beta$. Otherwise one of the ψ_{g_i}s would have calculated the group value for the entire block.
(2,3,4): Let the current point satisfy the formulas $G^P_{g_i}\langle\gamma_1, \ldots \gamma_k\rangle$, $\Gamma_l\langle\gamma_1, \ldots, \gamma_k\rangle$ and $G^F_{g_{j'}}\langle\gamma_1, \ldots, \gamma_k\rangle$. Let us assume $g_j = g_l.g_{j'}$. Then any point in the future will satisfy $G^P_{g'}\langle\gamma_1, ..., \gamma_k\rangle$ iff it also satisfies $G^F_{g_jg_ig'^{-1}}\langle\gamma_1, ...\gamma_k\rangle$. This let 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. □

Lemma 3.5 *The following translations are equivalent.*

(i) $\alpha_1 S(\alpha_2 \vee \alpha_3) \equiv \alpha_1 S\alpha_2 \vee \alpha_1 S\alpha_3$

(ii) $(\alpha_1 \wedge \alpha_2)S\alpha_3 \equiv \alpha_1 S\alpha_3 \wedge \alpha_2 S\alpha_3$

(iii) $\neg(a\mathsf{S}b) \equiv (\neg b\mathsf{S}\neg a) \vee \mathsf{H}\neg b$

(iv) $\neg G^P_g\langle\phi_1,...,\phi_k\rangle \equiv \bigvee_{g\neq g_i} G^P_{g_i}\langle\phi_1,...,\phi_k\rangle$

(v) $\alpha_1\mathsf{U}(\alpha_2 \vee \alpha_3) \equiv \alpha_1\mathsf{U}\alpha_2 \vee \alpha_1\mathsf{U}\alpha_3$

(vi) $(\alpha_1 \wedge \alpha_2)\mathsf{U}\alpha_3 \equiv \alpha_1\mathsf{U}\alpha_3 \wedge \alpha_2\mathsf{U}\alpha_3$

(vii) $\neg(a\mathsf{U}b) \equiv (\neg b\mathsf{U}\neg a) \vee \mathsf{G}\neg b$

(viii) $\neg G^F_g\langle\phi_1,...,\phi_k\rangle \equiv \bigvee_{g\neq g_i} G^F_{g_i}\langle\phi_1,...,\phi_k\rangle$

Proof. It is easy to see that the fourth and eigth statements are correct. The rest of the statements can be proved as shown in [6]. □

Lemma 3.4 and Lemma 3.5 let 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 propositional formulas. Let α, β and $\phi_1,...,\phi_k$ be formulas having the only modality being $a\mathsf{U}b$. Then $\alpha\mathsf{S}\beta$, $G^P_g\langle\phi_1,...,\phi_k\rangle$ can be separated.*

(ii) *Let a, b be propositional formulas. Let α, β and $\phi_1,...,\phi_k$ be formulas having the only modality being $a\mathsf{S}b$. Then $\alpha\mathsf{U}\beta$, $G^F_g\langle\phi_1,...,\phi_k\rangle$ 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 [6] shows how to separate the formula $\alpha\mathsf{S}\beta$. So let us consider the formula $G^P_g\langle\phi_1,...,\phi_k\rangle$. Let each ϕ_i be a boolean combination of $a\mathsf{U}b$ and propositions. To rewrite $G^P_g\langle\phi_1,...,\phi_k\rangle$ 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. □

Lemma 3.7 (i) *Let $a_1,...,a_k$ be propositional formulas. Let α, β and $\phi_1,...,\phi_k$ be formulas having only the modality $G^F_g\langle a_1,...,a_k\rangle$. Then $\alpha\mathsf{S}\beta$, $G^P_{g'}\langle\phi_1,...,\phi_k\rangle$ can be separated.*

(ii) *Let $a_1,...,a_k$ be propositional formulas. Let α, β and $\phi_1,...,\phi_k$ be formulas having only the modality $G^P_g\langle a_1,...,a_k\rangle$. Then $\alpha\mathsf{U}\beta$, $G^F_{g'}\langle\phi_1,...,\phi_k\rangle$ can be separated.*

Proof. Repeated application of Lemma 3.4 and Lemma 3.5 give us a separated formula. □

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

Lemma 3.8 (i) *Let $a_1,...,a_n, b_1,...,b_n$ be propositional formulas. Let α, β and $\phi_1,...,\phi_k$ be formulas having only the modality being $\forall i \leq n : U_i = a_i\mathsf{U}b_i$. Then $\alpha\mathsf{S}\beta$, $G^P_g\langle\phi_1,...,\phi_k\rangle$ can be separated.*

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

Proof. We prove (i) and claim that the proof for (ii) is similar. When $\psi = \alpha \mathsf{S} \beta$, this can be separated by the arguments of Gabbay. So let $\psi = G_g^P\langle \phi_1, ..., \phi_k\rangle$. Let $\{U_1, ..., U_n\}$ be the n Until formulas used in the ϕ_is. We first replace the Until formulas $U_1, ..., U_{n-1}$ by new propositions $p_1, ..., p_{n-1}$. 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. □

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

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

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

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

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

Proof. The proof is by combining the two Lemma 3.8 and Lemma 3.9. □

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 $a_1, ..., a_k, b_1, ..., b_k$ be propositional formulas. Let α be a formula such that the Future modalities are of the form $\forall i \leq n : U_i = a_i \mathsf{U} b_i s$ or $G_g^F\langle a_1, ..., a_k\rangle$. Then α can be separated.*

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

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

Lemma 3.12 *Let α be a formula such that no Past (Future) modality is nested inside a Future (Past) modality. Then α 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 ≥ 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_is by the Future (Past) modalites we replaced them with. Observe that we have reduced the Future (Past) depth. We repeat the above process until we get a separated formula. □

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

Theorem 3.13 *Let α be an LTLGRP (LTLMOD) formula. Then α can be separated.*

Proof. This involves induction on the alternation depth n of the formula α. $n = 1$ 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 α. Hence we can repeat the procedure until we get a formula which is separated. □

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

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

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 α be an LTLGRP (LTLMOD) formula. We can now write a first order logic formula, $\alpha'(x)$ on free variable x, such that it is equivalent to α. 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

$$\exists y((y < x) \wedge \phi(x, y)) \vee \phi(x, x) \vee (\exists y(y > x \wedge \phi(x, y)))$$

Now a formula of type $\exists y(y > x \wedge \phi(x,y))$ can be replaced by a pure future LTLGRP formula (since LTLGRP is expressively complete for FOGRP). Similarly we can replace pure past and pure present FOGRP formulas by pure past and pure present LTLGRP formulas. A similar proof can be given for group quantifiers too.
Now since LTLGRP (LTLMOD) is expressively complete for FOGRP (FOMOD) each of the separated formulas can be replaced with LTLGRP (LTLMOD) formulas. This gives us a separated formula.
($\Rightarrow$): We show that for an FOGRP (FOMOD) formula with one free variable we can give an equivalent LTLGRP (LTLMOD) formula. Let $P_1, ..., P_n$ 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 ϕ is of quantifier depth $< k$. We first remove x from ψ as follows. All subformulae of the form $x = x, x < x, x > x$ are replaced by $\top, \bot, \bot$ respectively. Now we rewrite ψ as follows (here $\boldsymbol{v} \in \{0,1\}^n$):

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

Here $\psi^v(x)$ replaces all occurrences of subformulas of the form $P_i(x)$ with $\top, \bot$ depending on v_i. Now the subformulas in each of ψ^v containing x will be of the form $x < z, x > z, x = z$, where z is some other variable in ψ. We remove these formulae by introducing three new unary predicates $R_<, R_>, R_=$ and replacing $x\ op\ z$ by $R_{op}(z)$ (op $:= \{<,>,=\}$). The resultant formula ψ^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 γ with new propositions $r_<, r_>, r_=$. We now write $\beta = \mathsf{P}\gamma \vee \gamma \vee \mathsf{F}\gamma$. Since LTLGRP formulas can be separated we can now separate β 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 $\top, \bot, \bot$ respectively. Similarly one can replace all the r_{op} formulae with $\top, \bot$ in the pure future, and pure present formulae.
Case 2, $Q = G_g$: So let $\psi^v = G_g y\langle\alpha_1(y), ..., \alpha_k(y)\rangle$. 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 ϕ_is with new propositions $r_<, r_>, r_=$.
Let us denote by $\Phi = \langle\phi_1, ..., \phi_k\rangle$. Then we can write $\beta = \bigvee_{i,j,l} G^P_{g_i}\Phi \wedge \Gamma_l(\Phi) \wedge G^F_{g_j}\Phi$, such that $g_i.g_l.g_j = g$ and $\psi^v = \beta$. Here $\Gamma_l(\Phi)$ is true only if $(\wedge_{j=1}^{l} \neg\phi_j) \wedge \phi_l$
We can now separate β 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 $\top, \bot, \bot$ respectively. Similarly one can replace all the r_{op} formulae with $\top, \bot$ 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 ψ formula will give us an LTLgrp formula which is equivalent to ψ. □

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 ψ. We now inductively built a FOgrp (FOmod) formula from ψ as follows. The translation, t : LTLgrp$\times\{x, y, z\}$ $\rightarrow$ FOgrp is inductively given. $t_x(\alpha)$, $t_y(\alpha)$, $t_z(\alpha)$ denotes $t(\alpha, x)$, $t(\alpha, y)$, $t(\alpha, z)$ respectively. For a formula α we give the translation t_x as follows.

LTLgrp	FOgrp
p	$P(x)$
$\alpha \vee \beta$	$t_x(\alpha) \vee t_x(\beta)$
$\neg\alpha$	$\neg t_x(\alpha(x))$
$\alpha \mathsf{U} \beta$	$\exists y\ (y > x) \wedge t_y(\beta(y)) \wedge \forall z\ (x < z < y) \implies t_z(\alpha(z))$
$\alpha \mathsf{S} \beta$	$\exists y\ (y < x) \wedge t_y(\beta(y)) \wedge \forall z\ (y < z < x) \implies t_z(\alpha(z))$
$G_g^F\langle\phi_1, ..., \phi_k\rangle$	$G_g y\langle(x < y) \wedge t_y(\phi_1(y)), ..., (x < y) \wedge t_y(\phi_k(y))\rangle$
$G_g^P\langle\phi_1, ..., \phi_k\rangle$	$G_g y\langle(x > y) \wedge t_y(\phi_1(y)), ..., (x > y) \wedge t_y(\phi_k(y))\rangle$

Clearly this translation uses only three variables and hence we get an equivalent FOgrp (FOmod) formula in three variables. □

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 FO^{2}grp).

Theorem 4.1 *UTLgrp is expressively complete for FO^{2}grp.*

Proof. The translation is recursive. For the atomic formulas, boolean combinations and existential formulas we follow the proof by Etessami et al [4]. We give here the translation for the group quantifier (A similar translation can be given for the modulo quantifiers). Let

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

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

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

Here τ^js are boolean propositional formula. γ_i^js 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 do not consider it here. α_i^js and β_i^js are formulas whose quantifier depth is less than the quantifier depth of $\phi(x)$. Moreover α_i^js and β_i^js are of type atomic or existential or group quantified. We first take out the α_i^js outside the group quantifier. For a vector $\boldsymbol{v} = (v_1, \ldots, v_{sk}) \in \{0,1\}^{sk}$ we define

$$\psi_{\boldsymbol{v}}^j(x,y) = \tau^j(\gamma_1^j(x,y), ..., \gamma_r^j(x,y), v_{j,1}, ..., v_{j,s}, \beta_1^j(y), ..., \beta_t^j(y))$$

Let ψ_g for a $g \in G$ be $G_g y \ \langle \psi_{\boldsymbol{v}}^1(x,y), ..., \psi_{\boldsymbol{v}}^k(x,y) \rangle$. Then we can rewrite ϕ as:

$$\phi(x) := \bigvee_{v \in \{0,1\}^{ks}} \left(\bigwedge_{j \leq k, i < s} \alpha_i^j(x) \Leftrightarrow v_{j,i} \right) \wedge \psi^g(x)$$

Observe that the γ_i^js 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, $o \in \Gamma$, γ_i^js will be evaluated to $\{T, F\}$. Let ψ_g^o be the formula got by replacing γ_i^j with T/F in $\psi_g(x)$ depending on the order o. Observe that x does not appear free in ψ_g^o. Thus we get

$$\phi(x) := \bigvee_{v \in \{0,1\}^{ks}} \left(\bigwedge_{j \leq k, i < s} \alpha_i^j(x) \Leftrightarrow v_{j,i} \right) \wedge \bigvee_{g_1 g_2 g_3 = g} \psi_{g_1}^{y<x} \wedge \psi_{g_2}^{x=y} \wedge \psi_{g_3}^{y>x}$$

Since formulas ψ_g^o do not contain free variables equivalent UTLgrp formulas exists. Formulas $\alpha_i^j(x)$ have equivalent UTLgrp formulas, since their quantifier depths are less than the quantifier depth of $\phi(x)$. Hence we get an UTLgrp formula equivalent to ϕ.
□

References

[1] Banieqbal, B. and H. Barringer, *Temporal logic with fixed points*, in: *Temporal Logic in Specification* (1987), pp. 62–74.
URL http://portal.acm.org/citation.cfm?id=647236.720405

[2] Barrington, D. A., N. Immerman and H. Straubing, *On uniformity within NC^1*, Journal of Computer and System Sciences **41** (1990), pp. 274–306.

[3] Baziramwabo, A., P. McKenzie and D. Thérien, *Modular temporal logic*, in: *Proc. 14th LICS* (1999), p. 344.

[4] Etessami, K., M. Y. Vardi and T. Wilke, *First-order logic with two variables and unary temporal logic*, Inf. Comput. **179** (2002), pp. 279–295.

[5] Gabbay, D., A. Pnueli, S. Shelah and J. Stavi, *On the temporal analysis of fairness*, in: *Proceedings of the Seventh ACM Symposium on Principles of Programming Languages*, ACM, 1980, pp. 163–173.

[6] Gabbay, D. M., *The declarative past and imperative future: Executable temporal logic for interactive systems*, in: *Temporal Logic in Specification* (1987), pp. 409–448.
URL `http://portal.acm.org/citation.cfm?id=647236.760143`

[7] Gabbay, D. M., I. Hodkinson and M. Reynolds, "Temporal Logics: Mathematical Foundations and Computational Aspects, Volume 1," Oxford University Press, Oxford, 1994.

[8] Henriksen, J. G. and P. Thiagarajan, *Dynamic linear time temporal logic*, Ann. Pure Appl. Logic **96** (1999), pp. 187–207.

[9] Hodkinson, I. M., *On Gabbay's temporal fixed point operator*, Theor. Comput. Sci. **139** (1995), pp. 1–25.

[10] Janin, D. and I. 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 (1996), pp. 263–277.
URL `http://portal.acm.org/citation.cfm?id=646731.703838`

[11] Kamp, J. A. W., "Tense Logic and the Theory of Linear Order," Ph.D. thesis, University of California, Los Angeles, CA (1968).

[12] Lodaya, K. and A. Sreejith, *LTL can be more succinct*, in: *Proc. 8th ATVA, Singapore*, LNCS **6252**, 2010, pp. 245–258.

[13] Pnueli, A., *The temporal logic of programs*, in: *focs77*, 1977, pp. 46–57.

[14] Prior, A. N., "Time and Modality," Oxford University Press, Oxford, 1956.

[15] R. McNaughton and S. Papert, "Counter-free Automata," MIT Press, Cambridge, USA, 1971.

[16] Stockmeyer, L. J. 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 (1973), pp. 1–9.
URL `http://doi.acm.org/10.1145/800125.804029`

[17] Straubing, H., D. Thérien and W. Thomas, *Regular languages defined with generalized quantifiers*, Inf. Comput. **118** (1995), pp. 389–301.

[18] Wolper, P., *Temporal logic can be more expressive*, Inf. Contr. **56** (1983), pp. 72–99.