The Journal of Symbolic Logic
Volume 00, Number 0, XXX 0000

FIRST ORDER QUANTIFIERS IN MONADIC SECOND ORDER LOGIC

H. JEROME KEISLER AND WAFIK BOULOS LOTFALLAH

Abstract. This paper studies the expressive power that an extra first order quantifier adds to a fragment of monadic second order logic, extending the toolkit of Janin and Marcinkowski [JM01].

We introduce an operation $exists_n(S)$ on properties S that says "there are n components having S". We use this operation to show that under natural strictness conditions, adding a first order quantifier word u to the beginning of a prefix class V increases the expressive power monotonically in u. As a corollary, if the first order quantifiers are not already absorbed in V, then both the quantifier alternation hierarchy and the existential quantifier hierarchy in the positive first order closure of V are strict.

We generalize and simplify methods from Marcinkowski [Mar99] to uncover limitations of the expressive power of an additional first order quantifier, and show that for a wide class of properties S, S cannot belong to the positive first order closure of a monadic prefix class W unless it already belongs to W.

We introduce another operation $alt(S)$ on properties which has the same relationship with the Circuit Value Problem as $reach(S)$ (defined in [JM01]) has with the Directed Reachability Problem. We use $alt(S)$ to show that $\Pi_n \not\subseteq FO(\Sigma_n)$, $\Sigma_n \not\subseteq FO(\Delta_n)$, and $\Delta_{n+1} \not\subseteq FOB(\Sigma_n)$, solving some open problems raised in [Mat98].

§1. Introduction. This paper studies the expressive power that an extra first order quantifier adds to a fragment of monadic second order logic.

Second order logic embodies many of the outstanding open problems in complexity theory. In [Fag74] Ronald Fagin showed that the class NP coincides with the class of properties expressible by existential second order sentences. Thus NP = co-NP if and only if the class of existential second order sentences is closed under negation. Stockmeyer [Sto77] subsequently extended Fagin's Theorem and showed that the polynomial hierarchy coincides with the second order quantifier alternation hierarchy, thus translating to logic the problem of the strictness of the polynomial hierarchy.

These hierarchy problems have been hard to attack. Fagin suggested studying monadic second order logic (MSO), a simplified fragment of full second order logic, in which second order quantifiers are only allowed over unary relations, i.e. subsets of the underlying universe. MSO was indeed tractable. In [Fag75] Fagin himself used Ehrenfeucht-Fraïssé games to show that existential MSO (called monadic NP) is not closed under negation, thus separating monadic NP from

This work was supported in part by the Vilas Trust Fund

0022-4812/00/0000-0000/$00.00

monadic co-NP. Matz and Thomas [MT97] showed that the monadic quantifier alternation hierarchy is strict. In particular, they showed that $\Sigma_n \subset B(\Sigma_n) \subset \Delta_{n+1} \subset \Sigma_{n+1}$, where B denotes Boolean closure of Σ_n. Their argument was based on growth rates of definable functions. In [Mat98] Matz used the growth argument to investigate the role of the positive first order closure in the monadic alternation hierarchy. Among other things, he showed that, on the class of finite graphs, $\Delta_{n+2} \not\subseteq FOB(\Sigma_n)$, $FO(\Sigma_n) \cap FO(\Pi_n) \not\subseteq B(\Sigma_n)$, and $FO(\Sigma_{n+1}) \cap FO(\Pi_{n+1}) \not\subseteq FOB(\Sigma_n)$, where FO denotes the positive first order closure, and FOB denotes the first order/Boolean closure.

Ajtai, Fagin, and Stockmeyer in [AFS98] and [AFS00] proposed closed monadic NP, in which first order quantifiers are freely mixed with monadic second order existential quantifiers, as the "right" monadic version of NP. They posed the problem of whether the corresponding hierarchy is strict. Marcinkowski [Mar99] showed that Directed Reachability is not in $FO(\Sigma_1)$, answering a question in [AFS98]. The tools of [AFS00] and [Mar99] were put in an abstract form by Janin and Marcinkowski in [JM01], to study the expressive power of fragments of MSO defined by prefix classes.

A (monadic) prefix class is a regular expression V built from the first order quantifiers $\forall, \exists$, monadic second order quantifiers $\forall, \exists$, and the Boolean closure operator $\oplus$. The logic $L(V)$ is the set of formulas built from words in V and finite conjunctions and disjunctions.

In [JM01] two operations on graph properties S were defined, *bool*(S) and *reach*(S). They call a prefix class *nontrivial* if it ends in $(\forall\exists\oplus)^\star$ and contains either an $\forall^\star$ or an $\exists^\star$. They proved the following results for nontrivial prefix classes V and W:

1) If both S and its complement are expressible in $L(V)$, but S is not expressible in $L(W)$, then *bool*(S) is expressible in $L(\exists\exists V)$ but not in $B(L(W))$.

2) If S is expressible in $L(V)$ but not in $L(W)$, then *reach*(S) is expressible in $L(\exists\forall\forall V)$ but not in $FO(L(W))$.

In this paper we introduce two more operations, $exists_n(S)$ and $alt(S)$, and prove the following results for arbitrary prefix classes V and W:

3) If S is expressible in $L(V)$ but not in $L(W)$, then for any $(n-1)$-tuple u of first order quantifiers, $exists_n(S)$ is expressible in $L(u\exists V)$ but not in $L(u\forall^\star W)$. (Lemma 3.2).

4) If V contains $\forall\forall$ and $\forall\exists$, and S is expressible in $L(V)$ but not in $L(W)$, then $alt(S)$ is expressible in $L(\exists\forall V) \cap L(\forall\exists V)$ but not in $FO(L(W))$. (Lemmas 6.6 and 6.7).

The operation $exists_n(S)$, introduced in Section 3, says "there are n components having property S". 3) above shows that this operation introduces an "existential hardness", so that adding a word $u\exists$ of first order quantifiers before a prefix class V increases the expressive power monotonically in u. As a corollary, if the first order quantifiers are not already absorbed in V, then both the first order quantifier alternation hierarchy and the first order existential quantifier hierarchy inside $FO(L(V))$ are strict. This improves a theorem in [KW73], where it is shown that (when V is empty) any two distinct first order quantifier words v, w express different sets of properties.

In Section 4 we simplify the proof of the result in [Mar99] that Directed Reachability is not expressible in $FO(\Sigma_1)$. The method is extended in Section 5 to show that for a wide class of properties S and for any monadic prefix class W, S cannot be expressed in $FO(L(W))$ unless it is already expressible in $L(W)$, and S cannot be expressed in $FOB(L(W))$ unless it is already expressed in $B(L(W))$.

In Section 6 we apply the ideas of Section 5 to define the operation *alt* and prove the result 4) above. *alt* has the same relationship with the Circuit Value Problem as *reach* has with the Directed Reachability Problem. The flexibility of *alt* allows us to strengthen some results and solve some open problems in [Mat98]. In particular, we show that $\Pi_n \not\subseteq FO(\Sigma_n)$, $\Sigma_n \not\subseteq FO(\Pi_n)$, and $\Delta_{n+1} \not\subseteq FOB(\Sigma_n)$.

§2. Basic definitions. For simplicity we only consider vocabularies containing one binary predicate E (for edge) and possibly several unary predicates and constant symbols. As usual, the logic has the equality symbol, $=$, as a built-in relation. We will not consider the case of logics with other built-in relations, such as linear order. All of the results in this paper hold for the class of finite models as well as the class of all models. That is, you can choose either one of the following two options at the outset, and stay with that option throughout the paper.

Finite option: Models are finite directed graphs with colors and distinguished vertices.

Infinite option: Models are arbitrary directed graphs with colors and distinguished vertices.

We use $\subseteq$ for inclusion, $\subset$ for strict inclusion, and $\not\subseteq$ for the negation of inclusion.

In this paper we consider only monadic second order extensions of first order logic. By a *logic* we will mean a set of monadic second order formulas which is positive Boolean closed, that is, closed under finite conjunctions and disjunctions. As usual, sentences are formulas with no free variables.

To clarify the different roles of universal and existential quantifiers, we assume that all negation signs are pushed inside. We shall sometimes view formulas as trees with nodes labelled by conjunction signs, disjunction signs, first order and monadic second order quantifications, and leafs labelled by literals, i.e. atomic and negation of atomic formulas.

Following [JM01], a *pattern* is a word over the alphabet $\{\forall, \exists, \mathbf{\forall}, \mathbf{\exists}, \oplus\}$, where $\forall, \exists$ are first order quantifiers, $\mathbf{\forall}, \mathbf{\exists}$ are monadic second order quantifiers, and $\oplus$ is the Boolean closure operator.

We let τ be a finite signature which contains at least one binary relation symbol and remains fixed throughout. For each signature τ and each pattern w, we define the logic $L(w)$ *supported by* w by induction as follows.

- The empty word supports the set of all quantifier-free MSO formulas with signature τ.
- $L(\forall w)$ is the positive Boolean closure of the set $L(w) \cup \{\forall x\varphi : \varphi \in L(w)\}$.
- $L(\exists w)$ is the positive Boolean closure of the set $L(w) \cup \{\exists x\varphi : \varphi \in L(w)\}$.
- $L(\mathbf{\forall} w)$ is the positive Boolean closure of the set $L(w) \cup \{\mathbf{\forall} X\varphi : \varphi \in L(w)\}$.

- $L(\exists w)$ is the positive Boolean closure of the set $L(w) \cup \{\exists X\varphi : \varphi \in L(w)\}$.
- $L(\oplus w)$ is the Boolean closure of $L(w)$, so $L(\oplus w) = B(L(w))$.

Example 1. *The prenex sentence* $({\scriptstyle\forall}\, x)({\scriptstyle\exists} y)(\forall\, Y)(E(x,y) \wedge Y(x))$ *is supported by the pattern* ${\scriptstyle\forall}\,{\scriptstyle\exists}\,\forall$ *, while the sentence* $({\scriptstyle\forall}\, x)(({\scriptstyle\exists} y)E(x,y) \wedge (\forall\, Y)Y(x))$ *is supported by the pattern* ${\scriptstyle\forall}\,{\scriptstyle\exists}\,\forall$ *as well as by* ${\scriptstyle\forall}\,\forall\,{\scriptstyle\exists}$*.*

We shall freely make use of regular expressions which do not contain union to denote classes of patterns, which we will call *prefix classes*. The logic $L(W)$ *supported by* a prefix class W is defined to be the union of the logics $L(w), w \in W$. Note that each $L(W)$ is positive Boolean closed.

Example 2. *The positive first order closure of a logic* $L(V)$ *is the logic*

$$FO(L(V)) = L(({\scriptstyle\forall}\,{\scriptstyle\exists})^\star V).$$

First order logic is the logic $FO = L(({\scriptstyle\forall}\,{\scriptstyle\exists})^\star)$*.*
Monadic NP *is the logic* $\Sigma_1 = L(\exists^\star({\scriptstyle\forall}\,{\scriptstyle\exists})^\star) = L((\exists\,{\scriptstyle\exists})^\star({\scriptstyle\forall}\,{\scriptstyle\exists})^\star)$*.*
Monadic co-NP *is* $\Pi_1 = L(\forall^\star({\scriptstyle\forall}\,{\scriptstyle\exists})^\star) = L((\forall\,{\scriptstyle\forall})^\star({\scriptstyle\forall}\,{\scriptstyle\exists})^\star)$*.*
Closed monadic NP *is* $L((\exists\,{\scriptstyle\forall}\,{\scriptstyle\exists})^\star)$*.*
Closed monadic co-NP *is* $L((\forall\,{\scriptstyle\forall}\,{\scriptstyle\exists})^\star)$*.*
$\Sigma_{n+1} = L(\exists^\star V) = L((\exists\,{\scriptstyle\exists})^\star V)$ *where* $\Pi_n = L(V)$*,*
$\Pi_{n+1} = L(\forall^\star V) = L((\forall\,{\scriptstyle\forall})^\star V)$ *where* $\Sigma_n = L(V)$*,*
$\Delta_n = \Sigma_n \cap \Pi_n$*.*

We do not allow regular expressions with unions, such as ${\scriptstyle\exists}({\scriptstyle\exists} \cup {\scriptstyle\forall})^2$, in the definition of a prefix class, because we do not have a corresponding Ehrenfeucht-Fraïssé game in Theorem 2.2 below.

The reader should be warned that for certain prefix classes W, it is not true that for every formula in $L(W)$ is equivalent to a prenex formula in $L(W)$, as the following example shows.

Example 3. *The sentence*

$${\scriptstyle\exists} x {\scriptstyle\exists} y(E(x,y) \wedge E(y,x)) \wedge {\scriptstyle\exists} x {\scriptstyle\exists} y(E(x,y) \wedge \neg E(y,x))$$

is supported by the pattern ${\scriptstyle\exists}\,{\scriptstyle\exists}$*, but there is no equivalent prenex sentence which is supported by* ${\scriptstyle\exists}\,{\scriptstyle\exists}$*. To support an equivalent prenex sentence one must go to the pattern* ${\scriptstyle\exists}\,{\scriptstyle\exists}\,{\scriptstyle\exists}\,{\scriptstyle\exists}$*.*

In this paper, it will always be understood that S denotes a property of (enriched) graphs, and that V and W denote prefix classes.

We shall identify a class of MSO sentences with the class of graph properties expressible by those sentences, where a graph property is the set of graphs having this property. Thus we write $S \in L(W)$ if the property S is expressible by a sentence in $L(W)$.

The *complement* of a graph property S is the class $\overline{S}$ of all graphs that do not have S. The *dual* $\overline{w}$ of a pattern w is the pattern obtained by switching ${\scriptstyle\exists}$ with ${\scriptstyle\forall}$ and $\exists$ with $\forall$. The dual of a prefix class W is the class $\overline{W} = \{\overline{w} : w \in W\}$. Note that $S \in L(W)$ if and only if $\overline{S} \in L(\overline{W})$. Thus, when we get an expressibility statement about S or W, we also get for free a dual statement about $\overline{S}$ or $\overline{W}$.

Since our convention is to push negations inside, the Boolean closure $B(L(W)) = L(\oplus W)$ of a logic is defined as the positive Boolean closure of $L(W) \cup L(\overline{W})$.

We recall from [JM01] that each pattern w naturally corresponds to an Ehrenfeucht-Fraïssé game between two players (Spoiler and Duplicator) played on a pair of (colored) graphs with distinguished points. For brevity, we will call a colored graph with distinguished points an *enriched graph.* Consider a pair $\mathcal{A}, \mathcal{B}$ of enriched graphs with the same signature.

The game w on the pair $\mathcal{A}, \mathcal{B}$ proceeds according to the following rules:

$w = {\scriptstyle\exists} v$ ($w = {\scriptstyle\forall} v$):
(1) Spoiler chooses a vertex $c \in \mathcal{A}$ ($d \in \mathcal{B}$).
(2) Duplicator chooses a vertex $d \in \mathcal{B}$ ($c \in \mathcal{A}$).
(3) Spoiler and Duplicator play v on the enriched pair $(\mathcal{A}, c), (\mathcal{B}, d)$.

$w = \exists v$ ($w = \forall v$): The game proceeds according to the following rules:
(1) Spoiler chooses a subset $C \subseteq \mathcal{A}$ ($D \subseteq \mathcal{B}$).
(2) Duplicator chooses a subset $D \subseteq \mathcal{B}$ ($C \subseteq \mathcal{A}$).
(3) Spoiler and Duplicator play v on the enriched pair $(\mathcal{A}, C), (\mathcal{B}, D)$.

$w = \oplus v$:
(1) Spoiler chooses either the pattern v or the dual pattern $\overline{v}$.
(2) Spoiler and Duplicator play the pattern chosen by Spoiler on $\mathcal{A}, \mathcal{B}$.

$w = \emptyset$ (the empty word): Game is over.

Duplicator wins the game $\emptyset$ on $\mathcal{A}, \mathcal{B}$ if and only if $\mathcal{A}$ and $\mathcal{B}$ satisfy the same atomic sentences. (In other words, the tuples of distinguished vertices $\mathbf{a}$, $\mathbf{b}$ are either empty or the mapping $a_i \mapsto b_i$ is an isomorphism from the colored subgraph generated by $\mathbf{a}$ to the colored subgraph generated by $\mathbf{b}$.)

We write $\mathcal{A} \rightharpoondown_w \mathcal{B}$ if Duplicator wins the game w on the pair $\mathcal{A}, \mathcal{B}$; otherwise we write $\mathcal{A} \not\rightharpoondown_w \mathcal{B}$.

Remark 2.1. *Let $v = \overline{w}$.*
(i) $L(\oplus v) = L(\oplus w)$.
(ii) $\mathcal{A} \rightharpoondown_v \mathcal{B}$ if and only if $\mathcal{B} \rightharpoondown_w \mathcal{A}$.
(iii) $\mathcal{A} \rightharpoondown_{\oplus w} \mathcal{B}$ if and only if $\mathcal{A} \rightharpoondown_w \mathcal{B}$ and $\mathcal{B} \rightharpoondown_w \mathcal{A}$.
(iv) $\mathcal{A} \rightharpoondown_{\oplus w} \mathcal{B}$ if and only if $\mathcal{B} \rightharpoondown_{\oplus w} \mathcal{A}$.

The following basic theorem clarifies the role of games (see, for example, [EF99]). It depends on the fact that the logic $L(w)$ is positive Boolean closed.

Theorem 2.2. *$S \in L(w)$ if and only if for all (enriched) graphs $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \in S$ and $\mathcal{A} \rightharpoondown_w \mathcal{B}$ implies $\mathcal{B} \in S$.* $\dashv$

Thus, to show that some property $S \notin L(w)$, we construct two graphs $\mathcal{A} \in S, \mathcal{B} \notin S$ and show that $\mathcal{A} \rightharpoondown_w \mathcal{B}$. Quite often, these graphs will be built from smaller graphs by means of some operations. Here is a simple lemma which is often used without explicit mention when working with such graphs.

Lemma 2.3. *If $\mathcal{A} \rightharpoondown_w \mathcal{B}$ then $\mathcal{A} \rightharpoondown_v \mathcal{B}$ for each substring v of w.*

Proof. Duplicator can win the v game by playing the w game, but choosing imaginary moves for Spoiler and making imaginary responses at times which are in w but not in v. $\dashv$

The following definition and easy lemma are essentially in [JM01].

DEFINITION 2.4. *If $\mathcal{A}$ is a graph, cone($\mathcal{A}$) is the graph formed by adding a new vertex r, called the root of cone($\mathcal{A}$), and adding a directed edge from r to each vertex of $\mathcal{A}$.*

If $\mathcal{A}_1, \dots, \mathcal{A}_n$ are graphs, $\mathcal{A}_1 \uplus \dots \uplus \mathcal{A}_n$ denotes the union of disjoint copies of cone($\mathcal{A}_1$) through cone($\mathcal{A}_n$). We also write $n\mathcal{A} = \mathcal{A} \uplus \dots \uplus \mathcal{A}$ (n times).

LEMMA 2.5. *(i) If $\mathcal{A} \rightharpoondown_w \mathcal{B}$, then cone($\mathcal{A}$) $\rightharpoondown_w$ cone($\mathcal{B}$).*

(ii) If π is a permutation on $\{1, \dots, n\}$ and $\mathcal{A}_i \rightharpoondown_w \mathcal{B}_{\pi(i)}$ for all $i = 1, \dots, n$, then $(\mathcal{A}_1 \uplus \dots \uplus \mathcal{A}_n) \rightharpoondown_w (\mathcal{B}_1 \uplus \dots \uplus \mathcal{B}_n)$.

PROOF. Like many results later on, this lemma is proved by induction on the pattern w. We mention here that in order to carry out such inductions, one needs to prove an analogous result for enriched graphs. For part (ii), define $(\mathcal{C}_1 \uplus \dots \uplus \mathcal{C}_n)$ when $C_1, \dots, C_n$ are disjoint enriched graphs whose signatures all have the same set k of colors but have disjoint sets d_i of distinguished elements. The signature of $(\mathcal{C}_1 \uplus \dots \uplus \mathcal{C}_n)$ has the set k of colors again, and its set of distinguished elements is the union $d_1 \cup \dots \cup d_n$. The interpretation of each color in the sum is the union of its interpretations in the C_i.

We remark that this proof, like many later proofs in this paper, needs Lemma 2.3 to deal with the fact that a first order move is made in just one $\mathcal{A}_i$ or $\mathcal{B}_i$ at a time. ⊣

§3. The power of a new first order quantifier. In this section we will define a simple operation on graph properties and inductively apply it to show that if $L(V) \neq L(\exists V)$ and $L(V) \neq L(\forall V)$, then the expressive power of the logics $L(uV)$ increase monotonically as the string of first order quantifiers u grows.

DEFINITION 3.1. *If $n > 0$, exists$_n(S)$ is the colored graph property with two new colors green and white, saying that the graph has at least n components, each of which is a cone of a white graph which has property S and has a green root.*

The reader may wonder why the new colors are introduced in this definition, since the root of a cone is already distinguished as the unique vertex with indegree zero. The reason is that it takes a universal quantifier to say that a vertex has indegree zero, and the new colors avoid this quantifier. Note that Part (i) of the next lemma would be false without the new colors. However, it can be easily fixed if we require that all words in V contain at least one universal first order quantifier.

LEMMA 3.2. *Let $n > 0$, and suppose that $u \in (\exists \cup \forall)^{n-1}$.*

(i) If $S \in L(V)$, then $exists_n(S) \in L(u\exists V)$.

(ii) If $S \notin L(W)$, then $exists_n(S) \notin L(u\forall^\star W)$.

PROOF. (i) The proof is by induction on n. The result is clear for $n = 1$. Assume the result holds for n. Let ψ be a sentence in $L(u\exists V)$ which expresses $exists_n(S)$. Let x be a new variable and let $\theta(x)$ be the formula obtained from ψ by replacing each existential quantifier from the outer $u\exists$ by the corresponding relativized quantifier $\exists y \neq x$. Then the sentence $\forall x\theta(x)$ expresses $exists_{n+1}(S)$, and belongs to $L(\forall u\exists V)$. The sentence

$\exists x[x$ is a green root of a white S-component and $\theta(x)]$
also expresses $exists_{n+1}(S)$ but belongs to $L(\exists u \exists V)$.

(ii) Let $w \in W$. Since $S \notin L(W)$, there are two graphs $\mathcal{A} \in S$, $\mathcal{B} \notin S$, such that Duplicator wins w on $\mathcal{A}, \mathcal{B}$. Given m, we need to show that $exists_n(S) \notin L(u\forall^m w)$. Let $\mathcal{C} = n\mathcal{A} \uplus (n+m-1)\mathcal{B}$ and $\mathcal{D} = (n-1)\mathcal{A} \uplus (n+m)\mathcal{B}$. Then $\mathcal{C} \in exists_n(S)$ and $\mathcal{D} \notin exists_n(S)$. We describe a winning strategy for Duplicator in the game $u\forall^m w$ on $\mathcal{C}, \mathcal{D}$.

In the u part of the game, Duplicator's first $n-1$ moves exactly mirror Spoiler's first $n-1$ moves. This is possible because whenever Spoiler moves in a new component on one side, there will still be a new component of the same kind on the other side. Spoiler's next m moves will choose vertices of $\mathcal{D}$. These moves can also be mirrored by Duplicator, because for each new component in $\mathcal{D}$ there will still be a new component of the same kind in $\mathcal{C}$. At this point there is at least one $\mathcal{A}$-component $\mathcal{A}_0$ of $\mathcal{C}$ and one $\mathcal{B}$-component $\mathcal{B}_0$ of $\mathcal{D}$ which have not been pebbled. The game position is given by a pair of enhanced graphs $(\mathcal{C}, \mathbf{c}), (\mathcal{D}, \mathbf{d})$ which satisfies the hypotheses of Lemma 2.5, where the permutation π matches $\mathcal{A}_0$ with $\mathcal{B}_0$, and matches each other component of $\mathcal{C}$ with an exact copy in $\mathcal{D}$. By Lemma 2.5, Duplicator can win w on $(\mathcal{C}, \mathbf{c}), (\mathcal{D}, \mathbf{d})$, and thus can win the original game $u\forall^m w$ on $\mathcal{C}, \mathcal{D}$. $\dashv$

The next corollary follows at once. Part (b) is the dual of part (a).

Corollary 3.3. *Let V and W be prefix classes such that $L(V) \not\subseteq L(W)$. Then for each $m \geq 0$,*

(a) $\bigcap\{L(u\exists V) : u \in (\exists \cup \forall)^m\} \not\subseteq \bigcup\{L(u\forall^\star W) : u \in (\exists \cup \forall)^m\}$.

(b) $\bigcap\{L(u\forall V) : u \in (\exists \cup \forall)^m\} \not\subseteq \bigcup\{L(u\exists^\star W) : u \in (\exists \cup \forall)^m\}$. $\dashv$

Note that part (a) says that there is a single property, namely $exists_m(S)$, which is expressible in every one of the logics $L(u\exists V)$, $u \in (\exists \cup \forall)^m$, and is not expressible in any of the logics $L(u\forall^\star W)$, $u \in (\exists \cup \forall)^m$. Following our convention, we do not write $L((\exists \cup \forall)^m \forall^\star V)$ because the expression $(\exists \cup \forall)^m \forall^\star V$ has a union, and does not correspond to an Ehrenfeucht-Fraïssé game in Theorem 2.2.

Corollary 3.3 can be sharpened as follows. Given a first order quantifier word u, let $F(u)$ be the set of all prefix classes obtained by replacing each $\exists$ in u by either $\exists$ or $\forall^\star$, and replacing each $\forall$ in u by either $\forall$ or $\exists^\star$. For example, after absorbing single quantifiers into starred quantifiers and omitting duplicates, we have

$$F(\exists\forall\exists) = \{\exists\forall\exists, \forall^\star\exists, \exists^\star, \exists\forall^\star, \forall^\star\exists^\star, \forall^\star, \exists^\star\forall^\star, \forall^\star\exists^\star\forall^\star\},$$
$$F(\forall\exists\exists) = \{\forall\exists\exists, \exists^\star, \forall^\star\exists, \forall\exists\forall^\star, \exists^\star\forall^\star\exists, \exists^\star\forall^\star, \forall^\star\}.$$

Corollary 3.4. *Suppose that $L(V) \not\subseteq L(W)$, and let $m \geq 0$, $u \in (\exists \cup \forall)^m$, and $U \in F(u)$. Then:*

(a) $L(u\exists V) \not\subseteq L(U\forall^\star W)$.

(b) $L(u\forall V) \not\subseteq L(U\exists^\star W)$.

Proof. We prove (a). The proof is by induction on m. When $m = 0$ the result follows from Corollary 3.3. Assume the result holds for all $n < m$, and let $u \in (\exists \cup \forall)^m$ and $U \in F(u)$. If $U = u$ then the result follows from Corollary 3.3.

Suppose $U \neq u$, and $\exists$ is replaced by $\forall^\star$ at the first place where U differs from u. Then we have $u = s\exists t$ and $U = s\forall^\star T$ where $T \in F(t)$. (One or both of s, t

can be empty). Then t has length less than m, and by inductive hypothesis, $L(t\exists V) \not\subseteq L(T\forall^\star W)$. Let $V' = t\exists V, W' = T\forall^\star W$. Then $L(V') \not\subseteq L(W')$. Using Corollary 3.3 again, we have $L(s\exists V') \not\subseteq L(s\forall^\star W')$. But $s\exists V' = s\exists t\exists V = u\exists V$ and $s\forall^\star W' = s\forall^\star T\forall^\star W = U\forall^\star W$, so $L(u\exists V) \not\subseteq L(U\forall^\star W)$ and the induction is complete.

The proof is similar in the case that $\forall$ is replaced by $\exists^\star$ at the first place where U differs from u, so (a) holds in all cases. (b) is the dual of (a). ⊣

Now let $V = W$, and consider the first order quantifier alternation hierarchy over V. This hierarchy is defined by

$$\Sigma^0_{2n}(V) = L((\exists^\star\forall^\star)^n V),\ \Sigma^0_{2n+1}(V) = L((\exists^\star\forall^\star)^n\exists^\star V),$$

$\Pi^0_m(V)$ is the dual of $\Sigma^0_m(V)$, and $\Delta^0_m(V) = \Sigma^0_m(V)\cap\Pi^0_m(V)$. The union of each of these hierarchies is the logic $FO(L(V)) = L((\forall\exists)^\star V)$, the *positive first order closure* of $L(V)$.

Corollary 3.5. *(i) If $L(\exists V) \neq L(V)$ and $L(\forall V) \neq L(V)$, then the hierarchy is strict, that is, the logics $\Sigma^0_n(V), \Pi^0_n(V), \Delta^0_n(V)$, $n = 1, 2, \ldots$ are all different.*

(ii) If $L(\exists V) \neq L(V)$ and $L(\forall V) = L(V)$, then for each $n > 0$,

$$\Pi^0_{2n-1}(V) \subset \Sigma^0_{2n-1}(V) = \Sigma^0_{2n}(V) \subset \Pi^0_{2n}(V) = \Pi^0_{2n+1}(V).$$

(iii) If $L(\exists V) = L(V)$ and $L(\forall V) \neq L(V)$, then for each $n > 0$,

$$\Sigma^0_{2n-1}(V) \subset \Pi^0_{2n-1}(V) = \Pi^0_{2n}(V) \subset \Sigma^0_{2n}(V) = \Sigma^0_{2n+1}(V).$$

(iv) If $L(\exists V) = L(V) = L(\forall V)$, then the hierarchy collapses to $L(V)$. ⊣

Proof. All the inclusions are clear. We use Corollary 3.4 to prove the strict inclusions. Suppose $L(\exists V) \neq L(V)$. Put $V' = \exists V$, so that $L(V') \not\subseteq L(V)$. Let $u = (\exists\forall)^{n-1}$, so $U = (\forall^\star\exists^\star)^{n-1} \in F(u)$ in the notation of 3.4. Then $L(u\exists V') \not\subseteq L(U\forall^\star V)$. We have $L(u\exists V') = L(u\exists\exists V) \subseteq \Sigma^0_{2n-1}(V)$ and $L(U\forall^\star V) = \Pi^0_{2n-1}(V)$, so

$$\Sigma^0_{2n-1}(V) \not\subseteq \Pi^0_{2n-1}(V).$$

Similarly, starting from $t = \forall u$ we get

$$\Pi^0_{2n}(V) \not\subseteq \Sigma^0_{2n}(V).$$

Part (ii) now follows, and we get (iii) by duality. Finally, (i) is proved by putting all of these non-inclusions together. ⊣

In the next corollary we consider the hierarchy $W, \exists W, \exists^2 W, \ldots$, which is a refinement of $\Sigma^0_1(W)$, and $W, \forall W, \forall^2 W, \ldots$, which is a refinement of $\Pi^0_1(W)$.

Corollary 3.6. *(i) If $L(W) \subset L(\exists W)$, then $L(\exists^n W) \subset L(\exists^{n+1} W)$ for each n.*

(ii) If $L(W) \subset L(\forall W)$, then $L(\forall^n W) \subset L(\forall^{n+1} W)$ for each n. ⊣

Proof. (i) Suppose $L(W) \subset L(\exists W)$. By Corollary 3.3 with $V = \exists W$, we have $L(\exists W) \subset L(\exists\exists V) = L(\exists\exists\exists W)$. If $L(\exists\exists W) = L(\exists W)$, then we would also have $L(\exists\exists\exists W) = L(\exists\exists W)$, contradicting $L(\exists W) \subset L(\exists\exists\exists W)$. Therefore we must have $L(\exists W) \subset L(\exists\exists W)$. The desired result now follows by induction. Part (ii) is the dual of (i). ⊣

Finally, we prove that if adding one first order quantifier increases the expressive power of $L(W)$, then the expressive power of $L(uW)$ is almost always increased by adding one quantifier anywhere within u, and is always increased by adding two quantifiers anywhere within u.

THEOREM 3.7. *Suppose that $L(W) \subset L(\exists W)$ and $L(W) \subset L(\forall W)$. Let $t, u \in (\exists \cup \forall)^\star$ be two first order quantifier words such that:*
(a) t is a proper substring of u, and
(b) u is not $t\forall$ or $t\exists$.
Then $L(tW) \subset L(uW)$.

PROOF. Let $V = \exists W$ and $Z = \forall W$, so $L(W) \subset L(V)$ and $L(W) \subset L(Z)$ by hypothesis.

Suppose first that $|u| = |t| + 1$. Then $t = rs$ and $u = rqs$ where $q \in \{\forall, \exists\}$. We give the proof when $q = \exists$. The case $q = \forall$ is similar. By hypothesis (b), the string s is nonempty, and is not of the form $\exists^n$. We may also position q at the right end of an $\exists$-block in u, so that s begins with an $\forall$.

Suppose $s = \forall$. Then $uW = r\exists\forall W = r\exists Z$, and by Corollary 3.3, $L(uW) = L(r\exists Z) \not\subseteq L(r\forall^\star W)$, so $L(tW) = L(r\forall W) \subset L(uW)$.

Now suppose $|s| = 2$. Then either $s = \forall\exists$ or $s = \forall\forall$. If $s = \forall\exists$, then $uW = r\exists\forall\exists W = r\exists\forall V$. By Corollary 3.3, $L(uW) = L(r\exists\forall V) \not\subseteq L(r\forall^\star\exists^\star W)$, so $L(tW) = L(r\forall\exists W) \subset L(uW)$. If $s = \forall\forall$, then $uW = r\exists\forall\forall W = r\exists\forall Z$, and $L(uW) = L(r\exists\forall Z) \not\subseteq L(r\forall^\star\exists^\star W)$, so $L(tW) = L(r\forall\forall W) \subset L(uW)$.

We next suppose that $|s| > 2$. Then either $s = x\exists\exists, s = x\exists\forall, s = x\forall\exists$, or $s = x\forall\forall$ for some nonempty string x which starts with an $\forall$. Form X by replacing each $\forall$ in x by $\exists^\star$ and replacing each $\exists$ in x by $\forall^\star$. Then X starts with an $\exists^\star$.

If $s = x\exists\exists$, then by Corollary 3.4,

$$L(uW) = L(r\exists x\exists\exists W) = L(r\exists x\exists V) \not\subseteq L(r\forall^\star X\forall^\star W),$$

and $L(tW) = L(rx\exists\exists W)$. By checking all cases, one can see that the string $x\exists\exists$ fits inside $\forall^\star X$; each block in x goes to the preceding block in $\forall^\star X$, and the final $\exists\exists$ goes to the last $\exists^\star$ in X. Therefore $L(tW) \subset L(uW)$.

If $s = x\exists\forall$, then Corollary 3.4 gives

$$L(uW) = L(r\exists x\exists\forall W) = L(r\exists x\exists Z) \not\subseteq L(r\forall^\star X\forall^\star W),$$

and $L(tW) = L(rx\exists\forall W)$. This time $x\exists\forall$ fits inside $\forall^\star X\forall^\star$, and again $L(tW) \subset L(uW)$. The cases $s = x\forall\exists$ and $s = x\forall\forall$ are similar.

Finally, we suppose that $|u| \geq |t|+2$. By adding terms to t we may assume that $|u| = |t|+2$. We need only consider the cases that $u = t\exists\exists$ and $u = t\forall\forall$, because in all other cases we can add one more term to t and satisfy hypotheses (a) and (b). If $u = t\exists\exists$ then by Corollary 3.3 we have $L(uW) = L(t\exists V) \not\subseteq L(t\forall^\star W)$, so $L(tW) \subset L(uW)$. The case $u = t\forall\forall$ is similar. This completes the proof. ⊣

QUESTION 3.8. *Is Theorem 3.7 still true without hypothesis (b)?*

For example, do the hypotheses of Theorem 3.7 imply that $L(\exists\forall^n W) \subset L(\exists\forall^{n+1} W)$, or even $L(\exists W) \subset L(\exists\forall W)$? Note that we always have $L(\exists\forall^n W) \subset L(\exists\forall^{n+2} W)$, so either $L(\exists\forall^n W) \subset L(\exists\forall^{n+1} W)$ or $L(\exists\forall^{n+1} W) \subset L(\exists\forall^{n+2} W)$.

QUESTION 3.9. *Suppose $L(\exists W) \not\subseteq L(\forall W)$ and $L(\forall W) \not\subseteq L(\exists W)$. If $u, v \in (\exists \cup \forall)^\star$ and $L(uW) \subseteq L(vW)$, must u be a subsequence of v?*

§4. Directed Reachability. In this section we will work with graphs $\mathcal{A}$ which have two distinguished points, called the *source* and the *sink*. A graph has the *Directed Reachability* property, $DIR.REACH$, if it contains a directed path from the source to the sink.

It was shown in [Mar99] that $DIR.REACH$ does not belong to the positive first order closure $FO(\Sigma_1)$ of Σ_1 =monadic NP. In this section we give a simpler proof of this fact. Since $DIR.REACH$ is not in monadic NP, a fact shown in [AF90] and already used in the proof given in [Mar99], the desired result easily follows from the next lemma.

LEMMA 4.1. *Let W be a prefix class such that $DIR.REACH \notin L(W)$. Then $DIR.REACH \notin FO(L(W))$.*

PROOF. The hypothesis can be viewed as asserting that for each $w \in W$, there exists two graphs $\mathcal{A}, \mathcal{B}$, such that $\mathcal{A}$ is in $DIR.REACH$, $\mathcal{B}$ is not, and Duplicator can win w on $\mathcal{A}, \mathcal{B}$. Let a_0, a_1 be the source and sink of $\mathcal{A}$, and b_0, b_1 be the source and sink of $\mathcal{B}$. It suffices to prove:
(1) $DIR.REACH \notin L(\forall W)$.
(2) $DIR.REACH \notin L(\exists W)$.

(1) Let $\mathcal{C} = \mathcal{A} + \mathcal{B}$ and $\mathcal{D} = \mathcal{B} + \mathcal{B}$. Here the "sum" of two graphs $(\mathcal{A} + \mathcal{B})$ is the graph obtained by connecting disjoint copies of $\mathcal{A}$ and $\mathcal{B}$ in "parallel". That is, we first form the union of a copy of $\mathcal{A}$, a disjoint copy of $\mathcal{B}$, and two new vertices c_0 (the new source) and c_1 (the new sink). Then we connect c_0 to both of the old sources a_0, b_0, and connect both of the old sinks a_1, b_1 to c_1. The sum has the same signature as the original graphs, with just the two distinguished vertices c_0, c_1.

It is clear that $\mathcal{C}$ is in $DIR.REACH$ and $\mathcal{D}$ is not. By Theorem 2.2, to prove that $DIR.REACH \notin L(\forall W)$ it is enough to show that $\mathcal{C} \rightharpoondown_{\forall w} \mathcal{D}$.

Duplicator wins $\forall w$ on $\mathcal{C}, \mathcal{D}$ as follows. Spoiler picks a vertex d in one of the copies of $\mathcal{B}$ in $\mathcal{D}$. Duplicator responds by picking the corresponding vertex c in the copy of $\mathcal{B}$ in $\mathcal{C}$. Lemma 2.5 still holds for sums of the form $\mathcal{A} + \mathcal{B}$, with a minor change in the proof to take care of the source and sink. It follows that $(\mathcal{C}, c) \rightharpoondown_w (\mathcal{D}, d)$, and thus $\mathcal{C} \rightharpoondown_{\forall w} \mathcal{D}$.

(2) Let $\mathcal{C} = \mathcal{A} \cdot \mathcal{A} + \mathcal{A} \cdot \mathcal{A}$ and $\mathcal{D} = \mathcal{A} \cdot \mathcal{B} + \mathcal{B} \cdot \mathcal{A}$. Here the "product" of two graphs $(\mathcal{A} \cdot \mathcal{B})$ is the graph that results from connecting $\mathcal{A}$ and then $\mathcal{B}$ in "series".

The product $\mathcal{A} \cdot \mathcal{B}$ is like the sum $\mathcal{A} + \mathcal{B}$ except for the connections between the new and old distinguished vertices. For the product we connect c_0 to a_0, a_1 to b_0, and b_1 to c_1.

Again, $\mathcal{C}$ is in $DIR.REACH$ and $\mathcal{D}$ is not. By Theorem 2.2, it now suffices to prove that $\mathcal{C} \rightharpoondown_{\exists w} \mathcal{D}$.

Duplicator can win $\exists w$ on $\mathcal{C}, \mathcal{D}$ with the following strategy. Spoiler's first move c is in $\mathcal{C}$. If c is in the left half of the product $\mathcal{A} \cdot \mathcal{A}$, Duplicator chooses the corresponding vertex d in the left half of the product $\mathcal{A}\cdot\mathcal{B}$ in $\mathcal{D}$. It is easily seen that $(\mathcal{A}\cdot\mathcal{A}, c) \multimap_w (\mathcal{A}\cdot\mathcal{B}, d)$, and again $(\mathcal{C}, c) \multimap_w (\mathcal{D}, d)$ and $\mathcal{C} \multimap_{\exists w} \mathcal{D}$.

If c is in the right half of $\mathcal{A}\cdot\mathcal{A}$, Duplicator chooses d in the right half of the product $\mathcal{B}\cdot\mathcal{A}$ in $\mathcal{D}$, and we have $(\mathcal{A}\cdot\mathcal{A}, c) \multimap_w (\mathcal{B}\cdot\mathcal{A}, d)$, and again $(\mathcal{C}, c) \multimap_w (\mathcal{D}, d)$ and $\mathcal{C} \multimap_{\exists w} \mathcal{D}$. ⊣

The above proof fully utilizes the concept of connecting graphs in parallel and in series, which already appeared in [Mar99].

§5. Limits on the power of a first order quantifier. We will now extend the method of the preceding section to give a general method of proving that some property S is not expressible by a sentence in $L(W)$.

The idea is to have two operations on graphs ("addition" and "multiplication") that play the roles of connecting graphs in "parallel" and in "series" with respect to Property S, such that the addition operation is commutative, and a winning strategy for Duplicator is congruent with respect to both operations.

We consider enhanced graphs with a fixed finite signature σ, and let k be the number of distinguished vertices. Our main tool will be the general notion of a disjoint binary operation $(\mathcal{A}, \mathcal{B}) \mapsto \mathcal{A} * \mathcal{B}$, where $\mathcal{A}, \mathcal{B}$, and $\mathcal{A} * \mathcal{B}$ are graphs with signature σ. Informally, a disjoint binary operation takes a disjoint union of $\mathcal{A}$ and $\mathcal{B}$, adds new distinguished vertices $c_1, \dots, c_k$, and adds edges between the old and new distinguished vertices in a way prescribed by a fixed graph $\mathcal{C}_0$ called the connector.

Definition 5.1. *A* connector *is a graph $\mathcal{C}_0$ which has $3k$ vertices $a_1, \dots, a_k, b_1, \dots, b_k, c_1, \dots, c_k$, such that the distinguished vertices are $c_1, \dots, c_k$, none of the pairs (a_i, a_j) or (b_i, b_j) are edges of $\mathcal{C}_0$, and none of the vertices a_i or b_j are colored.*

The disjoint operation *with connector $\mathcal{C}_0$ is the binary operation $(\mathcal{A}, \mathcal{B}) \mapsto \mathcal{A}*\mathcal{B}$ constructed as follows.*

Make copies of $\mathcal{A}$ and $\mathcal{B}$ with distinguished vertices $a_1, \dots, a_k$ and $b_1, \dots, b_k$ respectively, such that $\mathcal{A}, \mathcal{B}$, and the set $\{c_1, \dots, c_k\}$ are disjoint. Then the set of vertices of $\mathcal{A} * \mathcal{B}$ is the union of the sets of vertices of $\mathcal{A}, \mathcal{B}, \mathcal{C}_0$, the set of edges of $\mathcal{A} * \mathcal{B}$ is the union of the sets of edges of $\mathcal{A}, \mathcal{B}, \mathcal{C}_0$, each color in $\mathcal{A} * \mathcal{B}$ is the union of its values in $\mathcal{A}, \mathcal{B}, \mathcal{C}_0$, and the distinguished vertices of $\mathcal{A} * \mathcal{B}$ are $c_1, \dots, c_k$.

A disjoint operation $*$ is commutative *if and only if for all $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} * \mathcal{B}$ is isomorphic to $\mathcal{B} * \mathcal{A}$.*

Note that a disjoint operation $*$ is completely determined by its connector up to isomorphism. That is, if $\mathcal{A}$ is isomorphic to $\mathcal{A}'$, and $\mathcal{B}$ is isomorphic to $\mathcal{B}'$, then $\mathcal{A} * \mathcal{B}$ is isomorphic to $\mathcal{A}' * \mathcal{B}'$. Moreover, $*$ is commutative if and only if the mapping which sends each a_i to b_i and fixes each c_i is an automorphism of the connector $\mathcal{C}_0$.

Example 4. *In the proof of Lemma 4.1, the sum $\mathcal{A}+\mathcal{B}$ is a commutative disjoint operation, and the connector is the graph with vertices $\{a_0, a_1, b_0, b_1, c_0, c_1\}$, distinguished vertices c_0, c_1, and edges $(c_0, a_0), (c_0, b_0), (a_1, c_1), (b_1, c_1)$.*

In the same proof, the product $\mathcal{A} \cdot \mathcal{B}$ is a noncommutative disjoint operation. Its connector has the same vertices and distinguished points as above, but has the edges $(c_0, a_0), (a_1, b_0), (b_1, c_1)$.

EXAMPLE 5. *In the trivial case that $k = 0$ (where the signature σ has no distinguished vertices), there is only one disjoint operation. This operation, the disjoint union of a copy of $\mathcal{A}$ and a copy of $\mathcal{B}$, is commutative and its connector is the empty graph.*

We remark that commutative disjoint operations are not possible for logics with built-in linear order (e.g.as in [Imm99]), where it is more difficult to obtain lower bounds on the expressive power of a fragment.

PROPOSITION 5.2. *Let $\cdot$ be a disjoint operation, $+$ be a commutative disjoint operation, $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ be graphs with k distinguished vertices, and w be a pattern such that $\mathcal{A} \dashv_w \mathcal{B}$. Then:*

(i)
(a) $\mathcal{A} \cdot \mathcal{C} \dashv_w \mathcal{B} \cdot \mathcal{C}$ and
(b) $\mathcal{C} \cdot \mathcal{A} \dashv_w \mathcal{C} \cdot \mathcal{B}$,
(i.e. $\dashv_w$ is a congruence relation with respect to $\cdot$.)

(ii)
(a) $\mathcal{A} + \mathcal{A} \dashv_{\exists w} \mathcal{A} + \mathcal{B}$.
(b) $\mathcal{A} + \mathcal{B} \dashv_{\forall w} \mathcal{B} + \mathcal{B}$.

(iii)
(a) $(\mathcal{A} \cdot \mathcal{A}) + (\mathcal{A} \cdot \mathcal{A}) \dashv_{\exists w} (\mathcal{A} \cdot \mathcal{B}) + (\mathcal{B} \cdot \mathcal{A})$.
(b) $(\mathcal{A} \cdot \mathcal{B}) + (\mathcal{B} \cdot \mathcal{A}) \dashv_{\forall w} (\mathcal{B} \cdot \mathcal{B}) + (\mathcal{B} \cdot \mathcal{B})$.

PROOF. (i) (a). Duplicator can win the w game on $\mathcal{A} \cdot \mathcal{C}, \mathcal{B} \cdot \mathcal{C}$ as follows. When Spoiler moves in a copy of $\mathcal{C}$ or in the connector on either side, Duplicator moves in the same place on the other side. When Spoiler moves in the copy of $\mathcal{A}$ or $\mathcal{B}$, Duplicator follows her winning strategy for the w game on $\mathcal{A}, \mathcal{B}$. The proof of part (i) (b) is similar.

The proof of (ii) is essentially the same as the proof of Lemma 4.1. For (a), Spoiler must first move in a copy of $\mathcal{B}$ or the connector in the right side, and Duplicator moves in the same place in the copy of $\mathcal{B}$ or the connector in the left side. After that, Duplicator follows her w strategy. The proof of (b) is similar.

For (iii), Duplicator's first move must match Spoiler's first move; if Spoiler moves in the left half of a product, so must Duplicator, and if Spoiler moves in a copy of $\mathcal{A}$ ($\mathcal{B}$), so must Duplicator. After that, Duplicator uses her w strategy. ⊣

Note that commutativity of the +-operation is vital in the proof of each Part of this lemma. The (possible) non-commutativity of the $\cdot$-operation is the reason behind the necessary complexity of Part (iii).

THEOREM 5.3. *Let $\cdot$ be a disjoint operation and $+$ be a commutative disjoint operation. Suppose that*
(i) Whenever

$$A \in S, \mathcal{B} \notin S,$$

we have

$$\mathcal{A}+\mathcal{B}\in S,\ \mathcal{B}+\mathcal{B}\notin S,$$

and

$$(\mathcal{A}\cdot\mathcal{A})+(\mathcal{A}\cdot\mathcal{A})\in S,\ (\mathcal{A}\cdot\mathcal{B})+(\mathcal{B}\cdot\mathcal{A})\notin S.$$

Then for any prefix class W:
(a) If $S\notin L(W)$, then $S\notin FO(L(W))$.
(b) If $\overline{S}\notin L(W)$, then $\overline{S}\notin FO(L(W))$.
(c) If $S\notin B(L(W))$, then $S\notin FOB(L(W))$.

PROOF. (a) By Proposition 5.2, whenever $\mathcal{A}\multimap_w\mathcal{B}$, then
(1) $\mathcal{A}+\mathcal{B}\multimap_{\forall w}\mathcal{B}+\mathcal{B}$,
(2) $(\mathcal{A}\cdot\mathcal{A})+(\mathcal{A}\cdot\mathcal{A})\multimap_{\exists w}(\mathcal{A}\cdot\mathcal{B})+(\mathcal{B}\cdot\mathcal{A})$.

Now suppose $S\notin L(W)$. For each $w\in W$, $S\notin L(w)$, and by Theorem 2.2 there are graphs $\mathcal{A}\in S$, $\mathcal{B}\notin S$ such that $\mathcal{A}\multimap_w\mathcal{B}$.

By hypotheses (i), the left hand sides of (1) and (2) belong to S while the right hand sides do not. Thus by Theorem 2.2 again, $S\notin L(\forall w)$ and $S\notin L(\exists w)$. Therefore $S\notin L(\forall W)$ and $S\notin L(\exists W)$. It now follows by induction that $S\notin FO(L(W))$.

(b) This follows from Part (a) by a duality argument. If $\overline{S}\notin L(W)$, then $S\notin L(\overline{W})$. By Part (a), $S\notin FO(L(\overline{W}))$. Thus $\overline{S}\notin FO(L(W))$.

(c) Suppose $S\notin B(L(W))$. Let $V=\oplus W$. Then $B(L(W))=L(V)$ and $S\notin L(V)$. By Part (a), $S\notin FO(L(V))=FO(L(W))$. ⊣

COROLLARY 5.4. *Theorem 5.3 holds when the hypothesis (i) is replaced by the following simpler properties:*

(ii) $\mathcal{A}+\mathcal{B}\in S$ if and only if $\mathcal{A}\in S$ or $\mathcal{B}\in S$.
(iii) $\mathcal{A}\cdot\mathcal{B}\in S$ if and only if $\mathcal{A}\in S$ and $\mathcal{B}\in S$.

PROOF. It is easily seen that (ii) and (iii) imply hypothesis (i) of Theorem 5.3. ⊣

Some simple (and typical) examples of properties having addition and multiplication operations which satisfy Theorem 5.3 (i) are those dealing with the values of logical formulas, where addition and multiplication simply correspond to disjunction and conjunction.

We use "rooted" colored graphs to encode propositional formulas, with a different color for each proposition symbol and connective. By a *graphical formula* we mean a colored graph which encodes a formula in the following way. A graphical formula has one distinguished vertex, called the root, which has indegree 0 and represents the main symbol in the formula.

An atomic formula, which is just a predicate symbol by itself, is encoded by a single vertex which is a root with the corresponding color. The symbol $\vee$ is blue, and the symbol $\wedge$ is yellow. Thus, the graphical formula encoding $\phi\vee\psi$ has a blue root with two outgoing edges pointing to (the roots of) graphical formulas encoding ϕ and ψ. Similar encoding can be done if the main connective is $\wedge$ or $\neg$.

The colors serve as a convenient tool for classifying the vertices. However, they are not essential to the encoding mechanism and can be replaced if we wish by suitable "gadgets" connected to the classified vertices.

We extend this encoding in the obvious way to stronger languages, such as FO, SO, or MSO.

Now define the *disjunction* (*conjunction*) of graphical formulas $\mathcal{A}$ and $\mathcal{B}$, denoted by $\mathcal{A} \vee \mathcal{B}$ ($\mathcal{A} \wedge \mathcal{B}$), by forming the union of a copy of $\mathcal{A}$, a disjoint copy of $\mathcal{B}$, and a new root r, then coloring r blue (yellow), and connecting r to the roots of $\mathcal{A}$ and $\mathcal{B}$.

Proposition 5.5. *The operations of disjunction and of conjunction on graphical formulas are both commutative disjoint operations.* ⊣

We next observe that the disjunction and conjunction operations behave like addition and multiplication in Theorem 5.3 for a very broad class of properties S.

Definition 5.6. *Let L be a (positive Boolean closed) logic and M be any class of structures for L. A graph is in $SAT(L, M)$ if and only if it encodes a sentence in L which is true in some structure in M.*

For example, if L is propositional logic and M is the class of all propositional structures, then $SAT(L, M)$ is the Satisfiability Problem for propositional logic.

If L is first order logic with equality and the constant symbol $TRUE$, and M is the class of L-structures with universe $\{TRUE, FALSE\}$, then $SAT(L, M)$ is a version of the Quantified Satisfiability Problem for propositional logic.

Note that if the set M contains only one structure, then the stronger conditions Corollary 5.4 (ii), (iii) hold for $SAT(L, M)$. For example, if L is propositional logic and $M = \{A\}$ for a particular assignment A of truth values to propositional symbols, then $SAT(L, M)$ is the Circuit Value Problem for A.

Proposition 5.7. *Let L be a logic, M be a class of structures for L, and let $S = SAT(L, M)$. If $\mathcal{A} \in S$ and $\mathcal{B} \notin S$ then*

(i) $\mathcal{A} \vee \mathcal{B} \in S$ and $\mathcal{B} \vee \mathcal{B} \notin S$.

(ii) $\mathcal{A} \wedge \mathcal{A} \in S$ and $(\mathcal{A} \wedge \mathcal{B}) \vee (\mathcal{B} \wedge \mathcal{A}) \notin S$.

Proof. (i) is obvious. For (ii), note that if $\mathcal{A} \in S$ then $\mathcal{A} \wedge \mathcal{A} \in S$, and if $\mathcal{B} \notin S$ then $\mathcal{A} \wedge \mathcal{B} \notin S$. ⊣

Now putting Propositions 5.5 and 5.7 together with Theorem 5.3, the following corollary is immediate.

Corollary 5.8. *Let W be a prefix class, L be a logic, and M be any class of structures for L. Then $SAT(L, M)$ does not belong to $FO(L(W))$ unless it already belongs to $L(W)$.* ⊣

§6. A sharper version of *reach*. In [JM01] Janin and Marcinkowski defined the operation *reach* on properties. (The following definition, suggested to them by Sockmeyer, is somewhat simpler than their original definition.)

For any property S of graphs with a distinguished vertex, $reach(S)$ holds for a graph $\mathcal{G}$ with distinguished vertices s and t if and only if there is a directed path from s to t such that for every x on this path and every y not on the path with $E(x,y)$, the connected component of $\mathcal{G} \setminus \{x\}$ with y as a distinguished vertex has the property S.

Recall from the introduction that a prefix class is *nontrivial* if it ends with $(\forall\exists\oplus)^\star$ and contains either an $\forall^\star$ or an $\exists^\star$. They showed the following:

LEMMA 6.1. *[JM01] If V and W are nontrivial prefix classes and S is a property of graphs that belongs to $L(V)$ but not to $L(W)$, then $reach(S)$ belongs to $L(\exists\forall\forall V)$ but not to $FO(L(W))$.* ⊣

Their proof was an adaptation of Marcinkowski's proof in [Mar99] of the fact that Directed Reachability does not belong to the positive first order closure of monadic NP.

In this section we will apply the results of the previous section to define an analog of *reach* called *alt*, for which a sharper lemma can be proved.

We consider only enriched graphs with a distinguished vertex r (called the root) and the disjoint colors blue, yellow, green and white. Note that having four colors may be thought of the result of having two possibly overlapping unary predicates $B(x)$ and $Y(x)$, where x is blue means $B(x) \wedge \neg Y(x)$, x is yellow means $Y(x) \wedge \neg B(x)$, x is green means $B(x) \wedge Y(x)$, and x is white means $\neg B(x) \wedge \neg Y(x)$.

We call an enriched graph a *potential tree* if the following holds:

1) Each vertex has exactly one of the colors blue, yellow, green, or white.
2) r is nonwhite and has indegree 0.
3) Each nonwhite vertex has indegree at most 1.
4) Each white vertex has an incoming edge from at most one green vertex.
5) A blue or yellow vertex cannot have an outgoing edge to a white vertex.
6) Green and white vertices have outgoing edges only to white vertices.
7) If a, b are white vertices and there is an edge from a to b, then for each green vertex g, there is an edge from g to a if and only if there is an edge from g to b.

Thus in a potential tree, the connected component of the root is a tree with blue and yellow vertices possibly leading to green vertices, which then point to disjoint white graphs.

LEMMA 6.2. *The property* Potential Tree *is in $L(\forall\forall\forall) \cap L(\forall\exists\forall)$.*

PROOF. It is clear that each condition 1-7 can be expressed in $L(\forall\forall\forall)$. For $L(\forall\exists\forall)$, note that 4 and 7 can be replaced by:

4a) For each white a there exists g such that for each green h, $E(h,a)$ implies $h = g$.

7a) For each white a, if there is a green g with $E(g,a)$ then there is a green g such that $E(g,a)$ and for all b, $E(a,b)$ implies $E(g,b)$. ⊣

For each green vertex g in a potential tree, the *fruit* of g is (the subgraph generated by) the set $\mathcal{G} = \{x : E(g,x)\}$. g is then called the *root* of the fruit $\mathcal{G}$. Note that each fruit $\mathcal{G}$ is a white graph (if the reader finds white fruits to be unappetizing, he can replace them by white flowers). Any vertex which is connected to a vertex in $\mathcal{G}$ by an edge belongs to $\mathcal{G} \cup \{g\}$, and any two distinct fruits are disjoint. Also, the graph $\mathcal{G} \cup \{g\}$ with g as a distinguished vertex is $cone(\mathcal{G})$.

In a potential tree, an *alternating tree* is a set of vertices T satisfying:

1) $r \in T$.

2) If $x \in T$ and x is blue (indicating an "or" node), then there exists a vertex y such that $E(x, y)$ and $y \in T$.

3) If $x \in T$ and x is yellow (indicating an "and" node), then all vertices y with $E(x, y)$ belong to T.

4) T has no white vertices.

A *co-alternating tree* is a set T satisfying Conditions 1-4 with "blue" and "yellow" interchanged in Conditions 2 and 3, thus interchanging the roles of "or" and "and" nodes.

By a *fruit* of an alternating (co-alternating) tree T we mean the fruit of a green vertex which belongs to T.

The *connected part* of an alternating (co-alternating) tree T is the connected component of T that contains the root r. Note that for any alternating (co-alternating) tree T, the connected part of T is also an alternating (co-alternating) tree.

DEFINITION 6.3. *$alt(S)$ is the property of enriched graphs saying that the graph is a potential tree that contains an alternating tree T such that each fruit of T is an S-graph.*

A potential tree is said to be *trivial* if the root r is green. In a trivial potential tree, the root r has a fruit $\mathcal{G}$, the connected component $\mathcal{G} \cup \{r\}$ is $cone(\mathcal{G})$, and $T = \{r\}$ is both an alternating and a co-alternating tree. Thus for any graph property S, $cone(\mathcal{G})$ has $alt(S)$ if and only if $\mathcal{G} \in S$. Here is a simple application of Lemma 2.5.

LEMMA 6.4. *Let W be a prefix class. If $S \notin L(W)$ then $alt(S) \notin L(W)$.*

PROOF. Let $w \in W$. By Theorem 2.2, there are $\mathcal{A} \in S$ and $\mathcal{B} \notin S$ such that $\mathcal{A} \multimap_w \mathcal{B}$. By Lemma 2.5, $cone(\mathcal{A}) \multimap_w cone(\mathcal{B})$. As noted above, $cone(\mathcal{A}) \in alt(S)$ and $cone(\mathcal{B}) \notin alt(S)$. Then by Theorem 2.2 again, $alt(S) \notin L(W)$. $\dashv$

LEMMA 6.5. *Let $\mathcal{A}$ be a potential tree. $\mathcal{A}$ does NOT have $alt(S)$ if and only if $\mathcal{A}$ has a co-alternating tree T such that no fruit of T is an S-graph.*

PROOF. Let $n(\mathcal{A})$ be the cardinality of the set of nonwhite vertices in the connected component of the root r. The proof is by induction on $n(\mathcal{A})$. In this proof, it will be understood that all trees mentioned are connected.

For the basis step, suppose $n(\mathcal{A}) = 1$, and let $T = \{r\}$. If r is an "and" vertex, then T is an alternating tree with no fruits, so $\mathcal{A}$ has $alt(S)$, and there are no co-alternating trees.

If r is an "or" vertex, then T is a co-alternating tree with no fruits, and there are no alternating trees, so $\mathcal{A}$ does not have $alt(S)$. If r is green, then T is both alternating and co-alternating, and $\mathcal{A}$ has $alt(S)$ if and only if the fruit at r is an S-graph.

Now suppose that $n(\mathcal{A}) > 1$ and the lemma holds for every potential tree $\mathcal{B}$ such that $n(\mathcal{B}) < n(\mathcal{A})$. Then r is either an "and" vertex or an "or" vertex, and has indegree 0 and outdegree $k > 0$. Let $s_1, \ldots, s_k$ be the vertices connected to r by an edge. For each $i \leq k$, let $\mathcal{B}_i$ be the enhanced subgraph of $\mathcal{A}$ with root s_i consisting of all vertices which can be reached from s_i by a directed path. Then $n(\mathcal{B}_i) < n(\mathcal{A})$.

Suppose first that r is an "and" vertex. Note that T is a (connected) alternating tree in $\mathcal{A}$ if and only if $T \cap \mathcal{B}_i$ is an alternating tree in $\mathcal{B}_i$ for all $i \leq k$. Therefore $\mathcal{A}$ has $alt(S)$ if and only if $\mathcal{B}_i$ has $alt(S)$ for all $i \leq k$. Moreover, for each $i \leq k$, a set $U \subseteq \mathcal{B}_i$ is a co-alternating tree in $\mathcal{B}_i$ if and only if $U \cup \{r\}$ is a co-alternating tree in $\mathcal{A}$. Thus $\mathcal{A}$ has a co-alternating tree such that no fruit is an S-graph if and only if for some $i \leq k$, $\mathcal{B}_i$ has a co-alternating tree such that no fruit is an S-graph. Using the inductive hypothesis, it follows that $\mathcal{A}$ does not have $alt(S)$ if and only if $\mathcal{A}$ has a co-alternating tree such that no fruit is an S-graph.

The proof when r is an "or" vertex is similar. ⊣

Lemma 6.6. *Let W be a prefix class. Suppose that either $S \notin L(W)$ or $alt(S) \notin L(W)$. Then $alt(S) \notin FO(L(W))$.*

Proof. Recall that $FO(L(W)) = L((\forall\exists)^\star W)$. By Lemma 6.4, $alt(S) \notin L(W)$. $alt(S)$ has addition and multiplication operations on graphs, which connect two graphs with "or" and "and" roots respectively. The hypotheses of Theorem 5.3 are easily verified for these operations, and the result follows. ⊣

Lemma 6.7. *Suppose V is a prefix class such that some $v \in V$ contains $\forall\forall$ and $\forall\exists$ as substrings. For each natural number n:*
(i) If $S \in L(V)$, then $alt(S) \in L(\exists\forall V) \cap L(\forall\exists V)$.
(ii) If $S \in L(\exists^n V)$, then $alt(S) \in L(\exists^{n+1}\forall V)$.
(iii) If $S \in L(\forall^n V)$, then $alt(S) \in L(\forall^{n+1}\exists V)$.

Proof. (i) follows from (ii) and (iii) if we let $n = 0$.

(ii) We call $\mathcal{A}$ *good* if it has property S, and *bad* otherwise. Since $S \in L(\exists^n V)$, there is some $v \in V$ such that for all good $\mathcal{A}$ and bad $\mathcal{B}$, Spoiler wins $\exists^n v$ on $\mathcal{A}, \mathcal{B}$. Since V is directed, we may assume that v also contains the substrings $\forall\forall, \forall\exists$.

Let $\mathcal{C} \in alt(S)$ and $\mathcal{D} \notin alt(S)$. We first show how Spoiler wins $\exists^{n+1}\forall v$ on $\mathcal{C}, \mathcal{D}$. If $\mathcal{D}$ is not a potential tree, then by Lemma 6.2, Spoiler wins by using the moves $\forall\forall\forall$ in $\forall v$ to point out the problem in $\mathcal{D}$.

Otherwise, he starts by choosing in $\mathcal{C}$ an alternating tree X such that every fruit of X is good. Duplicator must respond with an alternating tree Y in $\mathcal{D}$. (If she doesn't, Spoiler can win the game using $\forall\forall$ against a bad "and" vertex, and $\forall\exists$ against a bad "or" vertex). Then at least one fruit $\mathcal{B}$ of Y is bad. Since all fruits of X are good, Spoiler can combine his winning strategies to play $\exists^n$ in each good fruit of X against the bad fruit $\mathcal{B}$ of Y. In other words, in each $\exists$-move, Spoiler plays the union of the subsets of the (good) fruits of X determined by his winning strategies against the fixed bad fruit $\mathcal{B}$. (Note that, since C is a potential tree, fruits with different roots are disjoint.) Spoiler then uses the $\forall$-move to pebble the root of $\mathcal{B}$. Duplicator must respond by pebbling the root of a (good) fruit $\mathcal{G}$ of X. Finally, Spoiler can now win the game by playing his winning strategy for v on the pair $\mathcal{G}, \mathcal{B}$.

(iii) We now show how Spoiler wins $\forall^{n+1}\exists v$ on $\mathcal{C}, \mathcal{D}$. Suppose first that $\mathcal{D}$ is not a potential tree. Spoiler can use the $\forall\exists$ moves to simulate an $\forall$ move $d \in \mathcal{D}$ by choosing the set $Y = \{d\} \subseteq \mathcal{D}$ and then choosing an element of Duplicator's response $X \subseteq \mathcal{C}$ (if Duplicator chooses X empty, Spoiler can easily win by choosing d with a later $\forall$ move). v contains the substring $\forall\forall$, so again Lemma 6.2 shows that Spoiler can win by pointing out the problem in $\mathcal{D}$.

Otherwise, Spoiler starts by choosing in $\mathcal{D}$ a co-alternating tree Y such that every fruit of Y is bad. Duplicator must respond with a co-alternating tree X in $\mathcal{C}$. (If X is not co-alternating, Spoiler wins as before, because v contains both $\forall\forall$ and $\forall\exists$ as substrings). From this point we argue as in part (ii) to complete the proof. ⊣

Let Σ_n, Π_n, Δ_n be the levels of the monadic second order quantifier hierarchy (defined in Section 2).

Corollary 6.8. *Let $n > 0$, let L be any of the logics Σ_n, Π_n or Δ_n, and W be a prefix class.*
(i) If $L \not\subseteq L(W)$, then $L \not\subseteq FO(L(W))$.
(ii) If $L \not\subseteq B(L(W))$, then $L \not\subseteq FOB(L(W))$.

Proof. (i) For the case $L = \Sigma_n$, suppose $S \in \Sigma_n \setminus L(W)$. By Lemma 6.6, $alt(S) \notin FO(L(W))$. For some m, $S \in L(\exists^m U)$ where $L(U) = \Pi_{n-1}$. By Lemma 6.7, $alt(S) \in L(\exists^{m+1}\forall U) \subseteq \Sigma_n$. The case $L = \Pi_n$ is similar.

Some care is needed in the case $L = \Delta_n$ because Δ_n is not a logic of the form $L(V)$, but is the intersection of two such logics, $\Delta_n = \Sigma_n \cap \Pi_n$. Suppose $\Delta_n \not\subseteq L(W)$ and let $S \in \Delta_n \setminus L(W)$. Again, $alt(S) \notin FO(L(W))$ by Lemma 6.6. For some m, $S \in L(\exists^m U) \cap L(\forall^m \overline{U})$, where $L(U) = \Pi_{n-1}$ as before. By Lemma 6.7,

$$alt(S) \in L(\exists^{m+1}\forall U) \cap L(\forall^{m+1}\exists \overline{U}) \subseteq \Delta_n.$$

(ii) follows by applying (i) to the prefix class $\oplus W$. ⊣

We do not know whether Lemma 6.7 holds for *reach* in place of *alt*, or whether there is a way to obtain Corollary 6.8 using *reach* instead of *alt*.

It was shown in [MT97] that $\Pi_n \not\subseteq \Sigma_n$, $\Sigma_n \not\subseteq \Pi_n$, and $\Delta_{n+1} \not\subseteq B(\Sigma_n)$. Combining these results with Corollary 6.8, we get the following corollary which solves some open problems from [Mat98].

Corollary 6.9. *In the monadic second order quantifier hierarchy,*
1) $\Pi_n \not\subseteq FO(\Sigma_n)$.
2) $\Sigma_n \not\subseteq FO(\Pi_n)$.
3) $\Delta_{n+1} \not\subseteq FOB(\Sigma_n)$. ⊣

§7. Conclusion and open problems. We introduced the operation $exists_n(S)$ saying "there are n components having S", and used it to show that if a single new first order existential (or universal) quantifier strictly increases expressive power, then additional new first order quantifiers continue to strictly increase expressive power. This implies the strictness of all the natural quantifier hierarchies in the positive first order closure of a fragment of monadic second order logic.

An important problem is to find a similar operation for the second order (monadic) quantifiers. This could be used to get lower bounds on the expressive power of second order fragments, and perhaps to improve the strictness results on the monadic hierarchy and solve the strictness problem for the closed monadic hierarchy.

We introduced an abstract concept of addition and multiplication of graphs. As an application we showed that for any logic L and any class of structures M for L, the set of sentences in L which are satisfiable in M is not expressible in the positive first order closure $FO(L(W))$ unless it is already expressible in $L(W)$.

We introduced another operation which converts a property S which is expressible in $L(V)$ but not in logic $L(W)$ to a new property $alt(S)$ which is expressible in both $L(\exists \forall V)$ and $L(\forall \exists V)$ but not in the positive first order closure $FO(L(W))$. This was applied to the monadic second order hierarchy, showing that $\Pi_n \not\subseteq FO(\Sigma_n)$ and $\Delta_{n+1} \not\subseteq FOB(\Sigma_n)$.

A related problem is to find operations which convert a property not expressible in $L(W)$ to a property not expressible in $L(\exists W)$ or $L(\forall W)$. A solution to this problem could give upper bounds on expressive power, and answer some outstanding open questions in monadic second order logic.

References:

[AF90] M. Ajtai and R. Fagin, Reachability is harder for directed than for undirected finite graphs, J. Symb. Logic 55 (1990), pp. 113-150.

[AFS98] M. Ajtai, R. Fagin, and L. Stockmeyer, The closure of monadic NP, (extended abstract of [AFS00]) **Journal of Computer and System Sciences**, (1998), pp. 309-318.

[AFS00] M. Ajtai, R. Fagin, and L. Stockmeyer, The closure of monadic NP, **Proc.of 13th STOC**, vol. 60 (2000), pp. 660-716.

[EF99] H-D. Ebbinghaus and J. Flum, Finite Model Theory, Second Edition. Springer 1999.

[Ehr61] A. Ehrenfeucht, An application of games to the completeness problem for formalized theories, **Fund. Math.**, vol. 49 (1961), pp. 129-141.

[Fag74] R. Fagin, Generalized first-order spectra and polynomial-time recognizable sets, **Complexity of Computation**, SIAM-AMS Proceedings, R.M. Karp, editor, vol. 7 (1974), pp. 43-73.

[Fag75] R. Fagin, Monadic generalized spectra, **Zeitschrift fuer Mathematische Logik und Grundlagen der Mathematik**, vol. 21 (1975), pp. 89-96.

[Imm99] N. Immerman, Descriptive Complexity, **Springer Verlag**, New York, 1999.

[JM01] D. Janin and J. Marcinkowski, Toolkit for first order extensions of monadic games, **Proceedings of STACS**, Springer, LNCS, (2001).

[KW73] H. J. Keisler and W. Walkoe, The diversity of quantifier prefixes, **J. Symbolic Logic** vol. 38 (1973), pp. 79-85.

[Mar99] J. Marcinkowski, Directed Reachability: From Ajtai-Fagin to Ehrenfeucht-Fraïssé games, **Proc. of the Annual Conference of the European Association of Computer Science Logic**, (CSL 99), Springer LNCS 1683, pp. 338-349.

[Mat98] O. Matz, First-order closure and the monadic second-order alternation hierarchy, Technical Report No. 9807, **Institut für Informatik und Praktische Mathematik**, Christian-Albrechts-Universität, Kiel, (May 1998).

[MT97] O. Matz and W. Thomas, The monadic quantifier alternation hierarchy over Graphs is infinite, **Proc. 12th IEEE Symposium on Logic in Computer Science**, (1997), pp. 236-244.

[Sto77] L. Stockmeyer, The polynomial time hierarchy, **Theoretical Computer Science**, vol. 3 (1977), pp. 1-22.

DEPARTMENT OF MATHEMATICS
UNIVERSITY OF WISCONSIN
480 LINCOLN DRIVE
MADISON WI 53706
E-mail: keisler@math.wisc.edu

DEPT. OF ENG. MATH. AND PHYS.
CAIRO UNIVERSITY
CAIRO, EGYPT 11451
E-mail: lotfalla@uwalumni.com