Separation with modulo predicates

Varun Ramanathan

Chennai Mathematical Institute

Joint work with Pascal Weil and Thomas Place at LaBRI, Summer 2016

DeLTA meeting, Paris, March 2018

Overview

Introduction
- Logic and languages
- Decision problems
- State of the art

Main problem

Proof
- Block abstraction
- Stable monoid
- Transfer theorem
- Enrichment
- EF games for modulo predicates

Conclusion

Logic and Languages

Logic and Languages

There is a strong link between formal languages and logical systems.

Logic and Languages

There is a strong link between formal languages and logical systems.

Rational languages $\longleftrightarrow$ Finite monoids $\longleftrightarrow$ MSO sentences

Star-free language $\longleftrightarrow$ Aperiodic monoids $\longleftrightarrow$ FO sentences

Logic and Languages

There is a strong link between formal languages and logical systems.

Rational languages $\longleftrightarrow$ Finite monoids $\longleftrightarrow$ MSO sentences

Star-free language $\longleftrightarrow$ Aperiodic monoids $\longleftrightarrow$ FO sentences

- Eg.: $\exists x(\mathbf{a}(x) \wedge (\forall y(x < y) \implies \mathbf{b}(y)))$ defines the language A^*ab^*

Fragments of **FO**

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.
Natural fragments?

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.
Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.
Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

syntactic restrictions $\rightarrow$ quantifier alternation: $\mathbf{\Sigma}_k : \underbrace{\exists^*\forall^* \ldots}_{k}(\varphi)$,

$\mathbf{\Pi}_k : \underbrace{\forall^*\exists^* \ldots}_{k}(\varphi)$
(φ quantifier free)

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.
Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

syntactic restrictions $\rightarrow$ quantifier alternation: $\mathbf{\Sigma}_k : \underbrace{\exists^*\forall^*\ldots}_{k}(\varphi)$,

$\mathbf{\Pi}_k : \underbrace{\forall^*\exists^*\ldots}_{k}(\varphi)$
(φ quantifier free)

or by limiting no. of variables: for example $FO^2[<]$

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution. Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

syntactic restrictions $\rightarrow$ quantifier alternation: $\mathbf{\Sigma}_k : \underbrace{\exists^*\forall^* \ldots}_{k}(\varphi)$,

$\mathbf{\Pi}_k : \underbrace{\forall^*\exists^* \ldots}_{k}(\varphi)$
(φ quantifier free)

or by limiting no. of variables: for example $\mathrm{FO}^2[<]$

or both? - $\mathbf{\Sigma}_1^{\mathrm{FO}^2}[<]$

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution.
Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

syntactic restrictions $\rightarrow$ quantifier alternation: $\mathbf{\Sigma}_k : \underbrace{\exists^*\forall^*\ldots}_{k}(\varphi)$,

$\mathbf{\Pi}_k : \underbrace{\forall^*\exists^*\ldots}_{k}(\varphi)$
(φ quantifier free)

or by limiting no. of variables: for example $\mathrm{FO}^2[<]$

or both? - $\mathbf{\Sigma}_1^{\mathrm{FO}^2}[<]$

In this way we get a plethora of fragments. Eg. - $\mathbf{\Sigma}_1[<, max]$, $\mathrm{FO}^2[<, +1]$

Fragments of **FO**

Fragment: formulae closed under $\wedge$, $\vee$ and quantifier free substitution. Natural fragments?

Considering local predicates: order $[<]$, successor $[+1]$, *min*, *max*

syntactic restrictions $\rightarrow$ quantifier alternation: $\mathbf{\Sigma}_k : \underbrace{\exists^*\forall^*\ldots}_{k}(\varphi)$,

$\mathbf{\Pi}_k : \underbrace{\forall^*\exists^*\ldots}_{k}(\varphi)$

(φ quantifier free)

or by limiting no. of variables: for example $\mathrm{FO}^2[<]$

or both? - $\mathbf{\Sigma}_1^{\mathrm{FO}^2}[<]$

In this way we get a plethora of fragments. Eg. - $\mathbf{\Sigma}_1[<, max]$, $\mathrm{FO}^2[<, +1]$

What about non-FO definable predicates?

Modulo predicates

Modulo predicates

Definition

For any $d \in \mathbb{N}$, $0 \leq i < d$ we have $\mathbf{mod}_i^d(x)$: $w, [x = j] \models \mathbf{mod}_i^d(x)$ iff $i \equiv j (\mathrm{mod}\ d)$

Modulo predicates

Definition

For any $d \in \mathbb{N}$, $0 \leq i < d$ we have $\mathbf{mod}_i^d(x)$: $w, [x = j] \models \mathbf{mod}_i^d(x)$ iff $i \equiv j (\mathrm{mod}\ d)$

$$\mathbf{MOD}^d = \bigcup\{\mathbf{mod}_i^d : i < d\}, \mathbf{MOD} = \bigcup \mathbf{MOD}^d$$

Modulo predicates

Definition

For any $d \in \mathbb{N}$, $0 \leq i < d$ we have $\mathbf{mod}_i^d(x)$: $w, [x = j] \vDash \mathbf{mod}_i^d(x)$ iff $i \equiv j (\text{mod } d)$

$$\mathbf{MOD}^d = \bigcup \{\mathbf{mod}_i^d : i < d\}, \mathbf{MOD} = \bigcup \mathbf{MOD}^d$$

$(aa)^* \rightarrow \forall x(\mathbf{a}(x) \wedge (max(x) \implies \mathbf{mod}_0^2(x)))$

Modulo predicates

Definition

For any $d \in \mathbb{N}$, $0 \leq i < d$ we have $\mathbf{mod}_i^d(x)$: $w, [x = j] \vDash \mathbf{mod}_i^d(x)$ iff $i \equiv j (\text{mod } d)$

$$\mathbf{MOD}^d = \bigcup\{\mathbf{mod}_i^d : i < d\}, \mathbf{MOD} = \bigcup \mathbf{MOD}^d$$

$(aa)^* \rightarrow \forall x(\mathbf{a}(x) \wedge (max(x) \implies \mathbf{mod}_0^2(x)))$

$\{w : |w|_a \text{ is even}\}$ is not definable in FO$[<, \text{MOD}]$.

Modulo predicates

Definition

For any $d \in \mathbb{N}$, $0 \leq i < d$ we have $\mathbf{mod}_i^d(x)$: $w, [x = j] \vDash \mathbf{mod}_i^d(x)$ iff $i \equiv j (\mathrm{mod}\ d)$

$$\mathbf{MOD}^d = \bigcup \{\mathbf{mod}_i^d : i < d\}, \mathbf{MOD} = \bigcup \mathbf{MOD}^d$$

$(aa)^* \rightarrow \forall x(\mathbf{a}(x) \wedge (max(x) \implies \mathbf{mod}_0^2(x)))$

$\{w : |w|_a \text{ is even}\}$ is not definable in FO[$<$, MOD].

FO[$<$, **MOD**] is more powerful than FO[$<$]

Decision problems

Decision problems

Let $\mathcal{F}$ be a fragment of **FO**.

Decision problems

Let $\mathcal{F}$ be a fragment of **FO**.

Membership

Given a regular language L, is it definable in $\mathcal{F}$?

Decision problems

Let $\mathcal{F}$ be a fragment of **FO**.

Membership

Given a regular language L, is it definable in $\mathcal{F}$?

Separation

Given L_1 and L_2 regular, does there exist L definable in $\mathcal{F}$ such that $L_1 \subseteq L$ and $L \cap L_2 = \emptyset$?

Adding modular predicates - State of the art

Adding modular predicates - State of the art

The membership and separation problems have been studied for a lot of different fragments.

For successor:

- For varieties, Straubing defined an operation $V \mapsto V * D$ based on the wreath product of monoids. This corresponds to adding successor to the fragment.

Adding modular predicates - State of the art

The membership and separation problems have been studied for a lot of different fragments.

For successor:

- For varieties, Straubing defined an operation $V \mapsto V * D$ based on the wreath product of monoids. This corresponds to adding successor to the fragment.
- Place and Zeitoun have defined a language operation $\mathscr{C} \to \mathscr{C} \circ \mathrm{SU}$ and proved that it amounts to adding successor

For modulo predicates

Adding modular predicates - State of the art

The membership and separation problems have been studied for a lot of different fragments.

For successor:

- For varieties, Straubing defined an operation $V \mapsto V * D$ based on the wreath product of monoids. This corresponds to adding successor to the fragment.
- Place and Zeitoun have defined a language operation $\mathscr{C} \to \mathscr{C} \circ \mathrm{SU}$ and proved that it amounts to adding successor

For modulo predicates

- Barrington et al proved membership decidability for $\mathrm{FO}[<, MOD]$

Adding modular predicates - State of the art

The membership and separation problems have been studied for a lot of different fragments.

For successor:

- For varieties, Straubing defined an operation $V \mapsto V * D$ based on the wreath product of monoids. This corresponds to adding successor to the fragment.
- Place and Zeitoun have defined a language operation $\mathscr{C} \rightarrow \mathscr{C} \circ \mathrm{SU}$ and proved that it amounts to adding successor

For modulo predicates

- Barrington et al proved membership decidability for $\mathrm{FO}[<, MOD]$
- Dartois and Paperman proved the decidability of membership of $\mathcal{F}[<, \mathbf{MOD}]$ for several fragments

Separability

Separability

Question: If separation is decidable for $\mathcal{F}$, is it decidable for $\mathcal{F}[\mathbf{MOD}]$?

Separability

Question: If separation is decidable for $\mathcal{F}$, is it decidable for $\mathcal{F}[\mathbf{MOD}]$?

Answer: Yes, if $\mathcal{F}$ has successor in its signature.

Separability

Question: If separation is decidable for $\mathcal{F}$, is it decidable for $\mathcal{F}[\mathbf{MOD}]$?

Answer: Yes, if $\mathcal{F}$ has successor in its signature.

Theorem

Let $\mathbb{S}$ be a logical signature containing $+1$. Let $\mathcal{F}[\mathbb{S}]$-separation be decidable. Then, $\mathcal{F}[\mathbb{S}, \mathbf{MOD}]$-separation is decidable.

Separability

Question: If separation is decidable for $\mathcal{F}$, is it decidable for $\mathcal{F}[\mathbf{MOD}]$?

Answer: Yes, if $\mathcal{F}$ has successor in its signature.

Theorem

Let $\mathbb{S}$ be a logical signature containing $+1$. Let $\mathcal{F}[\mathbb{S}]$-separation be decidable. Then, $\mathcal{F}[\mathbb{S}, \mathbf{MOD}]$-separation is decidable.

We apply this theorem to fragments

$$\mathsf{FO}[<,+], \mathsf{FO}^2[<,+], \mathbf{\Sigma}_1[<,+1], \mathbf{\Sigma}_2[<,+1], \mathbf{\Sigma}_3[<,+1], \mathcal{B}\mathbf{\Sigma}_1[<,+], \mathcal{B}\mathbf{\Sigma}_2[<,+]$$

Proof idea

Proof idea

The proof proceeds in the following way:

- $\mathscr{C}$ is the class of languages definable in $\mathcal{F}$

Proof idea

The proof proceeds in the following way:

- $\mathscr{C}$ is the class of languages definable in $\mathcal{F}$
- Define an operation $\mathscr{C} \to \|\mathscr{C}\|$ on languages

Proof idea

The proof proceeds in the following way:

- $\mathscr{C}$ is the class of languages definable in $\mathcal{F}$
- Define an operation $\mathscr{C} \to \|\mathscr{C}\|$ on languages
- Transfer theorem of separation from $\|\mathscr{C}\|$ to $\mathscr{C}$

Proof idea

The proof proceeds in the following way:

- $\mathscr{C}$ is the class of languages definable in $\mathcal{F}$
- Define an operation $\mathscr{C} \to \|\mathscr{C}\|$ on languages
- Transfer theorem of separation from $\|\mathscr{C}\|$ to $\mathscr{C}$
- Prove that $\|\mathscr{C}\|$ indeed corresponds to languages definable in $\mathcal{F}[\mathbf{MOD}]$

Block abstraction

Block abstraction

Given an alphabet A and number d, A_d^* words of length $\leq d$

Block abstraction

Given an alphabet A and number d, A^*_d words of length $\leq d$

Words in A^* can be cut into blocks

Block abstraction

Given an alphabet A and number d, A_d^* words of length $\leq d$

Words in A^* can be cut into blocks
$aaaabbabab \to [aaa][abb][aba][b]$

$$\mu_d : A^* \to A_d^*$$
$$a_1 a_2 \ldots a_n \mapsto [a_1 a_2 \ldots a_d][a_{d+1} a_{d+2} \ldots a_{2d}] \ldots [a_{kd+1} \ldots a_{kd+r}]$$

Block abstraction

Given an alphabet A and number d, A_d^* words of length $\leq d$

Words in A^* can be cut into blocks
$aaaabbabab \to [aaa][abb][aba][b]$

$$\mu_d : A^* \to A_d^*$$
$$a_1 a_2 \dots a_n \mapsto [a_1 a_2 \dots a_d][a_{d+1} a_{d+2} \dots a_{2d}] \dots [a_{kd+1} \dots a_{kd+r}]$$

$$\|\mathscr{C}\|_d = \{L \colon \exists K \in \mathscr{C} \text{ such that } L = \mu_d^{-1}(K)\}$$

Block abstraction

Given an alphabet A and number d, A_d^* words of length $\leq d$

Words in A^* can be cut into blocks
$aaaabbabab \rightarrow [aaa][abb][aba][b]$

$$\mu_d : A^* \rightarrow A_d^*$$
$$a_1 a_2 \ldots a_n \mapsto [a_1 a_2 \ldots a_d][a_{d+1} a_{d+2} \ldots a_{2d}] \ldots [a_{kd+1} \ldots a_{kd+r}]$$

$$\|\mathscr{C}\|_d = \{L \colon \exists K \in \mathscr{C} \text{ such that } L = \mu_d^{-1}(K)\}$$

Definition

The block abstraction of $\mathscr{C}$ is

$$\|\mathscr{C}\| = \bigcup_{d \in \mathbb{N}} \|\mathscr{C}\|_d$$

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stability index $\to$ least s such that $\eta(A^s) = \eta(A^{2s})$. (s exists because M is finite)

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stability index $\to$ least s such that $\eta(A^s) = \eta(A^{2s})$. (s exists because M is finite)

Definition

Stab($\mathcal{M}$), the stable monoid is $\eta(A^s) \cup \{e\}$

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stability index $\to$ least s such that $\eta(A^s) = \eta(A^{2s})$. (s exists because M is finite)

Definition

Stab($\mathcal{M}$), the stable monoid is $\eta(A^s) \cup \{e\}$

Now we look at the free monoid M^*.

Definition

$\overline{m} = m_1 m_2 \dots m_n \in M^*$ is stable if $m_1, m_2, \dots m_{n-1} \in \text{Stab}(\mathcal{M})$ and $m_n \in \eta(A^{<d})$

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stability index $\to$ least s such that $\eta(A^s) = \eta(A^{2s})$. (s exists because M is finite)

Definition

Stab($\mathcal{M}$), the stable monoid is $\eta(A^s) \cup \{e\}$

Now we look at the free monoid M^*.

Definition

$\overline{m} = m_1 m_2 \dots m_n \in M^*$ is stable if $m_1, m_2, \dots m_{n-1} \in \text{Stab}(\mathcal{M})$ and $m_n \in \eta(A^{<d})$

Definition

$\lfloor L \rfloor_{\mathcal{M}} = \{\overline{m} \colon \overline{m} \text{ is stable and } \beta(m) \in \eta(L)\}$

Stable monoid and well formed words

$\mathcal{M} = (M, \cdot)$, finite.

$\eta : A^* \to \mathcal{M}$ morphism to a finite monoid

Stability index $\to$ least s such that $\eta(A^s) = \eta(A^{2s})$. (s exists because M is finite)

Definition

Stab($\mathcal{M}$), the stable monoid is $\eta(A^s) \cup \{e\}$

Now we look at the free monoid M^*.

Definition

$\overline{m} = m_1 m_2 \ldots m_n \in M^*$ is stable if $m_1, m_2, \ldots m_{n-1} \in \text{Stab}(\mathcal{M})$ and $m_n \in \eta(A^{<d})$

Definition

$\lfloor L \rfloor_{\mathcal{M}} = \{\overline{m} \colon \overline{m}$ is stable and $\beta(m) \in \eta(L)\}$

Transfer theorem

Lemma

Let L_1 and L_2 be regular languages recognized by the same monoid $\mathcal{M}$ via a morphism η, whose stability index is s. Then, L_1 and L_2 are $\|\mathscr{C}\|_s$-separable if and only if $\lfloor L_1 \rfloor_{\mathcal{M}}$ and $\lfloor L_2 \rfloor_{\mathcal{M}}$ are $\mathscr{C}$-separable.

Modular enrichment

Now we will move on to the second part of the proof, which relates the enrichment of the fragment with modular predicates to the class transformation defined earlier.

Modular enrichment

Now we will move on to the second part of the proof, which relates the enrichment of the fragment with modular predicates to the class transformation defined earlier.

Enrichment result. Let $\mathscr{C}$ be the class of languages definable in $\mathcal{F}[\mathbb{S}]$ with $+1 \in \mathbb{S}$. Then, $\|\mathscr{C}\|$ is the class of languages definable in $\mathcal{F}[\mathbb{S}, \textbf{MOD}]$.

Proof of enrichment result

Showing $\|\mathscr{C}\|$ languages are definable in $\mathcal{F}[\mathbb{S}, \mathbf{MOD}]$ is easy.

Proof of enrichment result

Showing $\|\mathscr{C}\|$ languages are definable in $\mathcal{F}[\mathbb{S}, \textbf{MOD}]$ is easy.

It involves transforming a $\mathcal{F}[\mathbb{S}](A_d^*)$ describing a language K into a formula $\mathcal{F}[\mathbb{S}, \textbf{MOD}](A^*)$ which describes $\mu_d^{-1}(K)$

Proof of enrichment result

Showing $\|\mathscr{C}\|$ languages are definable in $\mathcal{F}[\mathbb{S}, \mathbf{MOD}]$ is easy.

It involves transforming a $\mathcal{F}[\mathbb{S}](A_d^*)$ describing a language K into a formula $\mathcal{F}[\mathbb{S}, \mathbf{MOD}](A^*)$ which describes $\mu_d^{-1}(K)$

We do this by describing blocks using the successor predicate.

Proof of enrichment result

Showing $\|\mathscr{C}\|$ languages are definable in $\mathcal{F}[\mathbb{S}, \mathbf{MOD}]$ is easy.

It involves transforming a $\mathcal{F}[\mathbb{S}](A_d^*)$ describing a language K into a formula $\mathcal{F}[\mathbb{S}, \mathbf{MOD}](A^*)$ which describes $\mu_d^{-1}(K)$

We do this by describing blocks using the successor predicate.

Eg: the following formula

$$\exists x([\mathbf{aba}](x))$$

over A^3 is transformed to

$$\exists x_1\, x_2\, x_3(\mathbf{mod}_0^3(x_1) \wedge (x_3 = x_2 + 1) \wedge (x_2 = x_1 + 1) \wedge \mathbf{a}(x_1) \wedge \mathbf{b}(x_2) \wedge \mathbf{a}(x_3))$$

Proof of enrichment result

For the other direction, it is sufficient to prove the result using an Ehrenfeucht-Fraïsse game argument for $\mathcal{F} = \mathsf{FO}$.

Proof of enrichment result

For the other direction, it is sufficient to prove the result using an Ehrenfeucht-Fraïsse game argument for $\mathcal{F} = \mathrm{FO}$.

To prove that if $L \notin \|\mathscr{C}\|_d$, then L is not definable in $\mathbf{FO}[<, +, \mathbf{MOD}^d]$.

Proof of enrichment result

For the other direction, it is sufficient to prove the result using an Ehrenfeucht-Fraïsse game argument for $\mathcal{F} = \text{FO}$.

To prove that if $L \notin \|\mathscr{C}\|_d$, then L is not definable in $\mathbf{FO}[<, +, \mathbf{MOD}^d]$.

If we prove for an arbitrary integer d that L is not definable in $\mathbf{FO}[<, +, \mathbf{MOD}^d]$, we are done.

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for **FO**$[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<,+]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<,+]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \dots x_i$ and $P_i(w_2) = y_1, y_2 \dots y_i$, then we proceed to the next round iff

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \dots x_i$ and $P_i(w_2) = y_1, y_2 \dots y_i$, then we proceed to the next round iff
  - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \to$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \dots x_i$ and $P_i(w_2) = y_1, y_2 \dots y_i$, then we proceed to the next round iff
  - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)
  - $x_i > x_j$ iff $y_i > y_j$ (equivalently $<$ and $=$)

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \ldots x_i$ and $P_i(w_2) = y_1, y_2 \ldots y_i$, then we proceed to the next round iff
  - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)
  - $x_i > x_j$ iff $y_i > y_j$ (equivalently $<$ and $=$)
  - For $\mathbf{MOD}^d$ - $x_i \equiv y_i (\text{mod } d)$

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<,+]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \ldots x_i$ and $P_i(w_2) = y_1, y_2 \ldots y_i$, then we proceed to the next round iff
    - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)
    - $x_i > x_j$ iff $y_i > y_j$ (equivalently $<$ and $=$)
    - For $\mathbf{MOD}^d$ - $x_i \equiv y_i (\text{mod } d)$
    - x_i and y_i have the same letter

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \ldots x_i$ and $P_i(w_2) = y_1, y_2 \ldots y_i$, then we proceed to the next round iff
  - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)
  - $x_i > x_j$ iff $y_i > y_j$ (equivalently $<$ and $=$)
  - For $\mathbf{MOD}^d$ - $x_i \equiv y_i (\text{mod } d)$
  - x_i and y_i have the same letter
- If D cannot pick such a position, S wins and game is terminated

Ehrenfeucht-Fraisse game

$G_k(w_1, w_2) \rightarrow$ k-round game for $\mathbf{FO}[<, +]$ for two words w_1 and w_2:

- Spoiler (S) and Duplicator (D) play
- Round i: S picks a word from w_1,w_2 and plays a position on that word
- D plays a position on the other word
- $P_i(w_1) = x_1, x_2 \ldots x_i$ and $P_i(w_2) = y_1, y_2 \ldots y_i$, then we proceed to the next round iff
  - $x_i = x_j + 1$ iff $y_i = y_j + 1$ (equivalently -1)
  - $x_i > x_j$ iff $y_i > y_j$ (equivalently $<$ and $=$)
  - For $\mathbf{MOD}^d$ - $x_i \equiv y_i (\bmod\ d)$
  - x_i and y_i have the same letter
- If D cannot pick such a position, S wins and game is terminated
- If D can successfully play for k rounds, he/she wins the game

Strategy transfer

If for every k, there exists $w_1 \in L$ and $w_2 \notin L$ such that D has a strategy on $G_k(w_1, w_2)$, then L is not definable in the fragment.

Strategy transfer

If for every k, there exists $w_1 \in L$ and $w_2 \notin L$ such that D has a strategy on $G_k(w_1, w_2)$, then L is not definable in the fragment.

For our purposes, this translates to the following sufficient condition:

Strategy transfer

If for every k, there exists $w_1 \in L$ and $w_2 \notin L$ such that D has a strategy on $G_k(w_1, w_2)$, then L is not definable in the fragment.

For our purposes, this translates to the following sufficient condition:

For w_1 and w_2, if $G(\mu_d(w_1), \mu_d(w_2))$ and $G'(w_1, w_2)$ are the $\mathbf{FO}[<, +]$ and $\mathbf{FO}[<, +, \mathbf{MOD}^d]$ EF games, and if Duplicator has a winning strategy for G, then Duplicator has a winning strategy for G'.

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$ababbabba$

$ababbababba$

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$ababbabba$

$ababbababba$

- Round i - Spoiler plays x_i on w_1 in G'

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$ababbabba$

$ababbababba$

- Round i - Spoiler plays x_i on w_1 in G'

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$abababba$

$ababbababba$

- Round i - Spoiler plays x_i on w_1 in G'
- Simulate Spoiler playing position $\lfloor x_i/d \rfloor$ on $\mu_d(w_1)$ in G

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$abababba$

$ababbababba$

- Round i - Spoiler plays x_i on w_1 in G'
- Simulate Spoiler playing position $p_i = \lfloor x_i/d \rfloor$ on $\mu_d(w_1)$ in G
- Get position q_i played by Duplicator according to his winning strategy in G

Strategy Transfer

For $G(\mu_d(w_1), \mu_d(w_2))$

$[aba][bab][ba]$

$[aba][bba][bab][ba]$

For $G'(w_1, w_2)$

$ababbabba$

$ababbababba$

- Round i - Spoiler plays x_i on w_1 in G'
- Simulate Spoiler playing position $p_i = \lfloor x_i/d \rfloor$ on $\mu_d(w_1)$ in G
- Get position q_i played by Duplicator according to his winning strategy in G
- Play position $y_i = (q_i * d + x_i (\text{mod } d))$ on $\mu_d(w_2)$ in G'. Then move to round $i+1$

Conclusion and closing remarks

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.
- The proof techniques will work for any fragment of first order logic provided it has successor.

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.
- The proof techniques will work for any fragment of first order logic provided it has successor.
- If we have successor then in some sense we are getting modular separation "for free" because:

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.
- The proof techniques will work for any fragment of first order logic provided it has successor.
- If we have successor then in some sense we are getting modular separation "for free" because:
  - Use of successor to describe blocks of letters, effectively abstracting modularity

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.
- The proof techniques will work for any fragment of first order logic provided it has successor.
- If we have successor then in some sense we are getting modular separation "for free" because:
  - Use of successor to describe blocks of letters, effectively abstracting modularity
  - Language transformation is defined generically

Conclusion and closing remarks

- Trivial to prove that Duplicator wins using the strategy described earlier.
- The proof techniques will work for any fragment of first order logic provided it has successor.
- If we have successor then in some sense we are getting modular separation "for free" because:
  - Use of successor to describe blocks of letters, effectively abstracting modularity
  - Language transformation is defined generically
  - independence of strategy transfer on the number of rounds played in the EF game, reducing dependence of the proof on logical structure

Thank you!