Expressivity within second-order transitive-closure logic

Jonni Virtema

Hasselt University, Belgium
jonni.virtema@gmail.com

Joint work with Jan Van den Bussche and Flavio Ferrarotti

CSL 2018 – September 5th 2018

Descriptive Complexity

- Offers a machine independent description of complexity classes:
  - Time/Space used by a machine to decide a problem
    $\Rightarrow$ richness of the logical language needed to describe the problem.
- Complexity classes can/could be then separated by separating logics.
- Many characterisations are known:
  - Fagin's Theorem 1973: Existential second-order logic characterises NP.

Descriptive Complexity

- Offers a machine independent description of complexity classes:
  - Time/Space used by a machine to decide a problem
    $\Rightarrow$ richness of the logical language needed to describe the problem.
- Complexity classes can/could be then separated by separating logics.
- Many characterisations are known:
  - Fagin's Theorem 1973: Existential second-order logic characterises NP.

$$\text{"A graph is three colourable"} = \exists R \exists B \exists G \big(\text{"each node is labeled by exactly one colour"} \wedge \text{"adjacent nodes are always coloured with distinct colours"}\big)$$

Descriptive Complexity

- Offers a machine independent description of complexity classes:
  - Time/Space used by a machine to decide a problem
    $\Rightarrow$ richness of the logical language needed to describe the problem.
- Complexity classes can/could be then separated by separating logics.
- Many characterisations are known:
  - Fagin's Theorem 1973: Existential second-order logic characterises NP.
  - Least fixed point logic LFP characterises P on ordered structures.
  - First-order transitive closure logic characterises $\mathrm{NLOGSPACE}$ on ordered structures.
  - Second-order logic characterises the polynomial time hierarchy.
  - Second-order transitive closure logic characterises PSPACE.
  - ...

Second-order transitive closure logic SO(TC)

- Expressive declarative language – can express exactly all PSPACE properties.
- Can express step-wise defined properties in a natural and elegant manner.
  - Recursive properties of graphs: Determine whether a graph G can be built starting from some graph pattern G_p by some recursive procedure.
- Already the monadic fragment $\mathrm{MSO(TC)}$ can express many interesting properties:
  - On strings it characterises nondeterministic linear space.
  - Can express NP-complete problems (e.g., QBF).
  - Can express counting.

Transitive closure

The transitive closure $\mathrm{TC}(R)$ of a binary relation $R \subseteq A \times A$ is defined as follows

$$\mathrm{TC}(R) := \{(a, b) \in A \times A \mid \text{there exists a finite directed } R\text{-path from } a \text{ to } b\}.$$

In our setting A is set of tuples $(a_1, \dots a_n)$, where each a_i is either an *element* or a *relation* over some domain D.

Transitive closure

The transitive closure $\mathrm{TC}(R)$ of a binary relation $R \subseteq A \times A$ is defined as follows

$$\mathrm{TC}(R) := \{(a, b) \in A \times A \mid \text{there exists a finite directed } R\text{-path from } a \text{ to } b\}.$$

In our setting A is set of tuples $(a_1, \dots a_n)$, where each a_i is either an *element* or a *relation* over some domain D.

Transitive closure

Example

Let $G = (V, E)$ be an undirected graph. Then $(a, b) \in \mathrm{TC}(E)$ if a and b are in the same component of G, or equivalently, if there is a path from a to b in G.

Example

A graph $G = (V, E)$ has a Hamiltonian cycle (cycle that visits every node exactly once) if the following holds:

1. There is a relation $\mathcal{R}$ such that

$$(Z, z, Z', z') \in \mathcal{R} \quad \text{iff} \quad Z' = Z \cup \{z'\}, z' \notin Z \text{ and } (z, z') \in E.$$

2. The tuple $(\{x\}, x, V, y)$ is in the transitive closure of $\mathcal{R}$, for some $x, y \in V$ such that $(y, x) \in E$.

Transitive closure

Example

Let $G = (V, E)$ be an undirected graph. Then $(a, b) \in \mathrm{TC}(E)$ if a and b are in the same component of G, or equivalently, if there is a path from a to b in G.

Example

A graph $G = (V, E)$ has a Hamiltonian cycle (cycle that visits every node exactly once) if the following holds:

1. There is a relation $\mathcal{R}$ such that

$$(Z, z, Z', z') \in \mathcal{R} \quad \text{iff} \quad Z' = Z \cup \{z'\}, z' \notin Z \text{ and } (z, z') \in E.$$

2. The tuple $(\{x\}, x, V, y)$ is in the transitive closure of $\mathcal{R}$, for some $x, y \in V$ such that $(y, x) \in E$.

Definable relations

Let $\vec{x}$ and $\vec{y}$ be k-tuples of first-order variables, $\varphi(\vec{x}, \vec{y})$ an FO-formula, and $\mathfrak{A}$ a model.

- $\varphi(\vec{x}, \vec{y})$ defines a $2k$-ary relation on $\mathfrak{A}$.
- We consider this $2k$-ary relation as a binary relation over k-tuples.
- We denote by $\mathrm{BIN}\big(\varphi(\vec{x}, \vec{y})\big)$ this binary relation.

Logics with transitive closure operator

First-order transitive closure logic FO(TC):

$$\varphi ::= x = y \mid X(x_1, \ldots, x_k) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid [\mathrm{TC}_{\vec{x},\vec{x'}}\varphi](\vec{y}, \vec{y'}),$$

where $\vec{x}$, $\vec{x'}$, $\vec{y}$, and $\vec{y'}$ are tuples of first-order variables of the same length.

Semantics for the TC operator:

$$\mathfrak{A} \models_s [\mathrm{TC}_{\vec{x},\vec{x'}}\varphi](\vec{y}, \vec{y'}) \text{ iff } (s(\vec{y}), s(\vec{y'})) \in \mathrm{TC}\Big(\mathrm{BIN}\big(\varphi(\vec{x}, \vec{x'})\big)\Big)$$

Example

The sentence

$$\forall x \forall y\, x = y \vee [\mathrm{TC}_{z,z'} E(z, z')](x, y)$$

expresses connectivity of graphs (V, E).

Logics with transitive closure operator

First-order transitive closure logic $\mathrm{FO(TC)}$:

$$\varphi ::= x = y \mid X(x_1, \ldots, x_k) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid [\mathrm{TC}_{\vec{x},\vec{x'}}\varphi](\vec{y}, \vec{y'}),$$

where $\vec{x}$, $\vec{x'}$,$\vec{y}$, and $\vec{y'}$ are tuples of first-order variables of the same length.

Semantics for the TC operator:

$$\mathfrak{A} \models_s [\mathrm{TC}_{\vec{x},\vec{x'}}\varphi](\vec{y}, \vec{y'}) \text{ iff } \big(s(\vec{y}), s(\vec{y'})\big) \in \mathrm{TC}\Big(\mathrm{BIN}\big(\varphi(\vec{x}, \vec{x'})\big)\Big)$$

Example

The sentence

$$\forall x \forall y \, x = y \vee [\mathrm{TC}_{z,z'} E(z, z')](x, y)$$

expresses connectivity of graphs (V, E).

Logics with transitive closure operator

Second-order transitive closure logic SO(TC):

$$\varphi ::= x = y \mid X(x_1, \ldots, x_k) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid \exists Y \varphi \mid [\mathrm{TC}_{\vec{X},\vec{X}'}\varphi](\vec{Y}, \vec{Y}'),$$

where $\vec{X}$, $\vec{X}'$, $\vec{Y}$, and $\vec{Y}'$ are tuples of first-order and second-order variables of the same length and sort.

Semantics for the TC operator:

$$\mathfrak{A} \models_s [\mathrm{TC}_{\vec{X},\vec{X}'}\varphi](\vec{Y}, \vec{Y}') \text{ iff } \big(s(\vec{Y}), s(\vec{Y}')\big) \in \mathrm{TC}\Big(\mathrm{BIN}\big(\varphi(\vec{X}, \vec{X}')\big)\Big)$$

MSO(TC) is the fragment of SO(TC) in which all second-order variables have arity 1.

Logics with transitive closure operator

Second-order transitive closure logic SO(TC):

$$\varphi ::= x = y \mid X(x_1, \ldots, x_k) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x \varphi \mid \exists Y \varphi \mid [\mathrm{TC}_{\vec{X},\vec{X}'}\varphi](\vec{Y}, \vec{Y}'),$$

where $\vec{X}$, $\vec{X}'$, $\vec{Y}$, and $\vec{Y}'$ are tuples of first-order and second-order variables of the same length and sort.

Semantics for the TC operator:

$$\mathfrak{A} \models_s [\mathrm{TC}_{\vec{X},\vec{X}'}\varphi](\vec{Y}, \vec{Y}') \text{ iff } \big(s(\vec{Y}), s(\vec{Y}')\big) \in \mathrm{TC}\Big(\mathrm{BIN}\big(\varphi(\vec{X}, \vec{X}')\big)\Big)$$

MSO(TC) is the fragment of SO(TC) in which all second-order variables have arity 1.

The Härtig quantifier

$\mathfrak{A} \models_s \mathrm{H}xy(\varphi(x), \psi(y)) \Leftrightarrow$ the sets $\{a \in A \mid \mathfrak{A} \models_{s(x \mapsto a)} \varphi(x)\}$ and $\{b \in A \mid \mathfrak{A} \models_{s(y \mapsto b)} \psi(y)\}$ have the same cardinality

Example (The Härtig quantifier can be expressed in MSO(TC).)

Let $\psi_{\mathrm{decrement}}$ denote an FO-formula expressing that $s(X') = s(X) \setminus \{a\}$ and $s(Y') = s(Y) \setminus \{b\}$ for some $a \in s(X)$ and $b \in s(Y)$. Define

$$\psi_{\mathrm{ec}} := [\mathrm{TC}_{X,Y,X',Y'} \psi_{\mathrm{decrement}}](Z, Z', \emptyset, \emptyset).$$

- Now ψ_{ec} holds under s iff $|s(Z)| = |s(Z')|$.
- Therefore $\mathrm{H}xy(\varphi(x), \psi(y))$ is equivalent with the formula

$$\exists Z \exists Z' (\forall x(\varphi(x) \leftrightarrow Z(x)) \wedge \forall y(\psi(y) \leftrightarrow Z'(y)) \wedge \psi_{\mathrm{ec}}).$$

The Härtig quantifier

$\mathfrak{A} \models_s \mathrm{H}xy(\varphi(x), \psi(y)) \Leftrightarrow$ the sets $\{a \in A \mid \mathfrak{A} \models_{s(x \mapsto a)} \varphi(x)\}$ and $\{b \in A \mid \mathfrak{A} \models_{s(y \mapsto b)} \psi(y)\}$ have the same cardinality

Example (The Härtig quantifier can be expressed in MSO(TC).)

Let $\psi_{\mathrm{decrement}}$ denote an FO-formula expressing that $s(X') = s(X) \setminus \{a\}$ and $s(Y') = s(Y) \setminus \{b\}$ for some $a \in s(X)$ and $b \in s(Y)$. Define

$$\psi_{\mathrm{ec}} := [\mathrm{TC}_{X,Y,X',Y'}\psi_{\mathrm{decrement}}](Z, Z', \emptyset, \emptyset).$$

- Now ψ_{ec} holds under s iff $|s(Z)| = |s(Z')|$.
- Therefore $\mathrm{H}xy(\varphi(x), \psi(y))$ is equivalent with the formula

$$\exists Z \exists Z' (\forall x(\varphi(x) \leftrightarrow Z(x)) \land \forall y(\psi(y) \leftrightarrow Z'(y)) \land \psi_{\mathrm{ec}}).$$

The Härtig quantifier

$\mathfrak{A} \models_s \mathrm{H}xy(\varphi(x), \psi(y)) \Leftrightarrow$ the sets $\{a \in A \mid \mathfrak{A} \models_{s(x \mapsto a)} \varphi(x)\}$ and $\{b \in A \mid \mathfrak{A} \models_{s(y \mapsto b)} \psi(y)\}$ have the same cardinality

Example (The Härtig quantifier can be expressed in MSO(TC).)

Let $\psi_{\text{decrement}}$ denote an FO-formula expressing that $s(X') = s(X) \setminus \{a\}$ and $s(Y') = s(Y) \setminus \{b\}$ for some $a \in s(X)$ and $b \in s(Y)$. Define

$$\psi_{\text{ec}} := [\mathrm{TC}_{X,Y,X',Y'} \psi_{\text{decrement}}](Z, Z', \emptyset, \emptyset).$$

- Now ψ_{ec} holds under s iff $|s(Z)| = |s(Z')|$.
- Therefore $\mathrm{H}xy(\varphi(x), \psi(y))$ is equivalent with the formula

$$\exists Z \exists Z' \big(\forall x(\varphi(x) \leftrightarrow Z(x)) \wedge \forall y(\psi(y) \leftrightarrow Z'(y)) \wedge \psi_{\text{ec}}\big).$$

Hamiltonian cycle

Example

A graph $G = (V, E)$ has a Hamiltonian cycle if the following holds:

1. There is a relation $\mathcal{R}$ such that

$$(Z, z, Z', z') \in \mathcal{R} \quad \text{iff} \quad Z' = Z \cup \{z'\}, z' \notin Z \text{ and } (z, z') \in E.$$

2. The tuple $(\{x\}, x, V, y)$ is in the transitive closure of $\mathcal{R}$, for some $x, y \in V$ such that $(y, x) \in E$.

In the language of $\mathrm{MSO(TC)}$ this can be written as follows:

$$\exists xy \big(E(y, x) \wedge [\mathrm{TC}_{Z,z,Z',z'}\varphi](\{x\}, x, V, y)\big)$$

where $\varphi := \neg Z(z') \wedge \forall x\big(Z'(x) \leftrightarrow (Z(x) \vee z' = x)\big) \wedge E(z, z')$.

Hamiltonian cycle

Example

A graph $G = (V, E)$ has a Hamiltonian cycle if the following holds:

1. There is a relation $\mathcal{R}$ such that

$$(Z, z, Z', z') \in \mathcal{R} \quad \text{iff} \quad Z' = Z \cup \{z'\}, z' \notin Z \text{ and } (z, z') \in E.$$

2. The tuple $(\{x\}, x, V, y)$ is in the transitive closure of $\mathcal{R}$, for some $x, y \in V$ such that $(y, x) \in E$.

In the language of $\mathrm{MSO(TC)}$ this can be written as follows:

$$\exists xy \big(E(y, x) \wedge [\mathrm{TC}_{Z,z,Z',z'} \varphi](\{x\}, x, V, y) \big)$$

where $\varphi := \neg Z(z') \wedge \forall x \big(Z'(x) \leftrightarrow (Z(x) \vee z' = x) \big) \wedge E(z, z')$.

Descriptive complexity

Theorem (Harel and Peleg 84)

$\mathrm{SO(TC)}$ *characterises polynomial space* PSPACE.

Theorem (Immerman 87)

- *On finite ordered structures, first-order transitive-closure logic* $\mathrm{FO(TC)}$ *characterises nondeterministic logarithmic space* $\mathrm{NLOGSPACE}$.
- *On strings (word structures),* $\mathrm{SO(arity}\ k\mathrm{)(TC)}$ *captures* $\mathsf{NSPACE}(n^k)$.

In particular, on strings $\mathrm{MSO(TC)}$ characterises nondeterministic linear space NLIN.

Existential positive $\mathrm{SO(2TC)}$

$\exists\mathrm{SO(2TC)}$ is the syntactic fragment of $\mathrm{SO(TC)}$ in which

1. the existential quantifiers and the TC-operators occur only positively.
2. TC-operators bound only second-order variables.

Rosen noted (1999) that $\exists\mathrm{SO}$ collapses to existential first-order logic $\exists\mathrm{FO}$.

Theorem

The expressive powers of $\exists\mathrm{SO(2TC)}$ and $\exists\mathrm{FO}$ coincide.

Proof.

$$[\mathrm{TC}_{\vec{X},\vec{X}'}\exists x_1 \ldots \exists x_n \theta](\vec{Y},\vec{Y}') \text{ and } [\mathrm{TC}^{\leq k}_{\vec{X},\vec{X}'}\exists x_1 \ldots \exists x_n \theta](\vec{Y},\vec{Y}'),$$

where θ is quantifier free FO-formula, are equivalent for large enough k.
(Note that k is independent of the model and depends only on the formula.) □

Existential positive $\mathrm{SO(2TC)}$

$\exists\mathrm{SO(2TC)}$ is the syntactic fragment of $\mathrm{SO(TC)}$ in which

1. the existential quantifiers and the TC-operators occur only positively.
2. TC-operators bound only second-order variables.

Rosen noted (1999) that $\exists\mathrm{SO}$ collapses to existential first-order logic $\exists\mathrm{FO}$.

Theorem

The expressive powers of $\exists\mathrm{SO(2TC)}$ *and* $\exists\mathrm{FO}$ *coincide.*

Proof.

$$[\mathrm{TC}_{\vec{X},\vec{X}'} \exists x_1 \ldots \exists x_n \theta](\vec{Y}, \vec{Y}') \text{ and } [\mathrm{TC}^{\leq k}_{\vec{X},\vec{X}'} \exists x_1 \ldots \exists x_n \theta](\vec{Y}, \vec{Y}'),$$

where θ is quantifier free FO-formula, are equivalent for large enough k.
(Note that k is independent of the model and depends only on the formula.) □

Corridor tiling problem

The *corridor tiling problem* is the following PSPACE-complete decision problem (Chlebus 86):

Input: An instance $P = (T, H, V, \vec{b}, \vec{t}, n)$, where

- T is a finite collection of tiles,
- H and V are the horizontal and vertical constraints for tiling,
- $\vec{b}$ and $\vec{t}$ are n-tuples of tiles.

Output: Does there exists a tiling of width n having $\vec{b}$ as the bottom row and $\vec{t}$ as the top row of tiles?

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $2\mathrm{TC}[\forall\mathrm{FO}]$ *is* PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

□

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $2\mathrm{TC}[\forall\mathrm{FO}]$ is PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

- s on $\{1, \ldots, n\}$ encodes the horizontal incidence relation of the tiling.
- We construct two rows of tiling on top of each other:
  - $Z_1, \ldots Z_k$ encodes the tiling of the lower row,
  - $Z'_1, \ldots Z'_k$ encodes the tiling of the upper row,

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $\mathrm{2TC}[\forall\mathrm{FO}]$ *is* PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

$$\varphi_H := \forall xy\big(s(x,y) \to \bigvee_{(i,j)\in H} Z'_i(x) \wedge Z'_j(y)\big),$$

□

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $2\mathrm{TC}[\forall\mathrm{FO}]$ *is* PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

$$\varphi_V := \forall x \bigvee_{(i,j) \in V} Z_i(x) \wedge Z'_j(x)$$

□

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $\mathrm{2TC}[\forall\mathrm{FO}]$ *is* PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

$$\varphi_T := \text{every point } i \text{ is labelled with exactly one } Z_i^j,$$

□

Complexity of model checking

Theorem

Combined complexity of model checking for monadic $2\mathrm{TC}[\forall\mathrm{FO}]$ *is* PSPACE*-complete.*

Proof.

Hardness follows from corridor tiling. Input: $(T, H, V, \vec{b}, \vec{t}, n)$.
Let s be a successor relation on $\{1, \ldots, n\}$ and $X_1, \ldots X_k, Y_1, \ldots, Y_k$ monadic second-order variables (corresponding to tiles) that are used to encode $\vec{b}$ and $\vec{t}$.

$$\varphi_H := \forall xy\big(s(x,y) \rightarrow \bigvee_{(i,j)\in H} Z_i'(x) \wedge Z_j'(y)\big), \quad \varphi_V := \forall x \bigvee_{(i,j)\in V} Z_i(x) \wedge Z_j'(x)$$

$\varphi_T :=$ every point i is labelled with exactly one Z_i',

The formula $[\mathrm{TC}_{\vec{Z},\vec{Z}'}\, \varphi_T \wedge \varphi_H \wedge \varphi_V](\vec{X}, \vec{Y})$ describes proper tiling. □

MSO(TC) and counting

- *Counter variables* μ and ν on $\mathfrak{A}$ range over $\{0, \ldots, |A|\}$.
- Assume a supply of k-ary *numeric predicates* $p(\mu_1, \ldots, \mu_k)$.
  - Intuitively relations over natural numbers such as the table of multiplication.
  - Similar to generalised quantifiers; a k-ary numeric predicate is a set $Q_p \subseteq \mathbb{N}^{k+1}$ of $k+1$-tuples of natural numbers.
  - When evaluating a k-ary numeric predicate $p(\mu_1, \ldots, \mu_k)$, the numeric predicate Q_p accesses also the cardinality of the domain of the model.

MSO(TC) and counting

Definition

The syntax of CMSO(TC) extends the syntax of MSO(TC) as follows:

$$\varphi ::= \quad \mu = \#\{x : \varphi\} \mid p(\mu_1, \ldots, \mu_k) \mid \exists \mu \varphi \mid [\mathrm{TC}_{\vec{X}, \vec{X}'} \varphi](\vec{Y}, \vec{Y}'),$$

where $\vec{X}$, $\vec{X}'$, $\vec{Y}$, and $\vec{Y}'$ may also include counter variables.

Semantics:

$\mathfrak{A} \models_s \mu = \#\{x : \varphi\}$ iff $s(\mu)$ equals the cardinality of $\{a \in A \mid \mathfrak{A} \models_{s(x \mapsto a)} \varphi\}$.

$\mathfrak{A} \models_s p(\mu_1, \ldots, \mu_k)$ iff $(|A|, s(\mu_1), \ldots, s(\mu_k)) \in Q_p$

$\mathfrak{A} \models_s \exists \mu \varphi$ iff there exists $i \in \{0, \ldots, |A|\}$ such that $\mathfrak{A} \models_{s(\mu \mapsto i)} \varphi$.

MSO(TC) and counting

Definition

The syntax of CMSO(TC) extends the syntax of MSO(TC) as follows:

$$\varphi ::= \quad \mu = \#\{x : \varphi\} \mid p(\mu_1, \ldots, \mu_k) \mid \exists \mu \varphi \mid [\mathrm{TC}_{\vec{X}, \vec{X}'} \varphi](\vec{Y}, \vec{Y}'),$$

where $\vec{X}$, $\vec{X}'$, $\vec{Y}$, and $\vec{Y}'$ may also include counter variables.

Semantics:

$\mathfrak{A} \models_s \mu = \#\{x : \varphi\}$ iff $s(\mu)$ equals the cardinality of $\{a \in A \mid \mathfrak{A} \models_{s(x \mapsto a)} \varphi\}$.

$\mathfrak{A} \models_s p(\mu_1, \ldots, \mu_k)$ iff $\big(|A|, s(\mu_1), \ldots, s(\mu_k)\big) \in Q_p$

$\mathfrak{A} \models_s \exists \mu \varphi$ iff there exists $i \in \{0, \ldots, |A|\}$ such that $\mathfrak{A} \models_{s(\mu \mapsto i)} \varphi$.

Counting in NLOGSPACE

Definition (NLOGSPACE numeric predicates)

We restrict to predicates Q_p for which the membership $(n_0, \ldots, n_k) \in Q_p$ can be decided in NLOGSPACE, when the numbers $n_0, \ldots, n_k$ are given in unary.

Example

Let k be a natural number, $X, Y, Z, X_1, \ldots, X_n$ monadic second-order variables. The following numeric predicates are decidable in NLOGSPACE:

- $\mathfrak{A} \models_s \mathrm{size}(X, k)$ iff $|s(X)| = k$,
- $\mathfrak{A} \models_s \times(X, Y, Z)$ iff $|s(X)| \times |s(Y)| = |s(Z)|$,
- $\mathfrak{A} \models_s +(X_1, \ldots, X_n, Y)$ iff $|s(X_1)| + \cdots + |s(X_n)| = |s(Y)|$.

Counting in NLOGSPACE

Definition (NLOGSPACE numeric predicates)

We restrict to predicates Q_p for which the membership $(n_0, \ldots, n_k) \in Q_p$ can be decided in NLOGSPACE, when the numbers $n_0, \ldots, n_k$ are given in unary.

Example

Let k be a natural number, $X, Y, Z, X_1, \ldots, X_n$ monadic second-order variables. The following numeric predicates are decidable in NLOGSPACE:

- $\mathfrak{A} \models_s \mathrm{size}(X, k)$ iff $|s(X)| = k$,
- $\mathfrak{A} \models_s \times(X, Y, Z)$ iff $|s(X)| \times |s(Y)| = |s(Z)|$,
- $\mathfrak{A} \models_s +(X_1, \ldots, X_n, Y)$ iff $|s(X_1)| + \cdots + |s(X_n)| = |s(Y)|$.

Counting in NLOGSPACE

Proposition (Immerman 87)

For every k-ary numeric predicate Q_p decidable in NLOGSPACE *there exists a formula φ_p of* FO(TC) *over $\{suc, x_1, \ldots, x_k\}$,*

$$\mathfrak{A} \models_s p(\mu_1, \ldots, \mu_k) \text{ iff } \mathfrak{B} \models_t \varphi_p,$$

where $B = \{0, 1, \ldots, |A|\}$, $t(suc)$ is the successor relation of B, and $t(x_i) = s(\mu_i)$, for $1 \leq i \leq k$.

$\mathrm{MSO(TC)}$ (without order) simulates $\mathrm{FO(TC)}$ with order

Natural numbers i are simulated by sets of cardinality i.
Recall that $\mathrm{MSO(TC)}$ can express the Härtig quantifier!

The translation $^+ : \mathrm{FO(TC)} \to \mathrm{MSO(TC)}$ is defined as follows:

- For ψ of the form $x_i = x_j$, define $\psi^+ := \mathrm{H}xy\big(X_i(x), X_j(y)\big)$.
- For ψ of the form $suc(x_i, x_j)$, define $\psi^+ := \exists z\Big(\neg X_i(z) \wedge \mathrm{H}xy\big(X_i(x) \vee x = z, X_j(y)\big)\Big)$.
- All other cases: identity, where x_is replaced by X_is.

$\mathrm{MSO(TC)}$ (without order) simulates $\mathrm{FO(TC)}$ with order

Natural numbers i are simulated by sets of cardinality i.
Recall that $\mathrm{MSO(TC)}$ can express the Härtig quantifier!

The translation $^+ : \mathrm{FO(TC)} \rightarrow \mathrm{MSO(TC)}$ is defined as follows:

- For ψ of the form $x_i = x_j$, define $\psi^+ := \mathrm{H}xy\big(X_i(x), X_j(y)\big)$.
- For ψ of the form $suc(x_i, x_j)$, define $\psi^+ := \exists z\Big(\neg X_i(z) \wedge \mathrm{H}xy\big(X_i(x) \vee x = z, X_j(y)\big)\Big)$.
- All other cases: identity, where x_is replaced by X_is.

MSO(TC) simulates CMSO(TC)

In MSO(TC) counter variables are treated as set variables.
Define a translation $^* : \mathrm{CMSO(TC)} \to \mathrm{MSO(TC)}$.

- For an NLOGSPACE numeric predicate Q_p and ψ of the form $p(\mu_1, \dots, \mu_k)$, define

$$\psi^* := \varphi_p^+(\mu_1, \dots, \mu_k),$$

  where $^+$ is the translation defined in the previous slide and φ_p is the defining FO(TC) formula of Q_p.
- For ψ of the form $\mu = \#\{x \mid \varphi\}$, the translation ψ^* is $\mathrm{H}xy(\varphi^*(x), \mu(y))$.
- All other cases: identity

Order invariant MSO

- We compare order-invariant MSO with MSO(TC).
- In order-invariant MSO, we have an access to an ordering of the model, but the truth of formulas should not depend on which order is present.
- E.g., even cardinality is expressible in order-invariant MSO.

Order invariant MSO and MSO(TC)

Example

Consider the class
$$\mathcal{C} = \{\mathfrak{A} \mid |A| \text{ is a prime number}\}$$
of $\emptyset$-structures. The language of prime length words over some unary alphabet is not regular and thus it follows via Büchi's theorem that $\mathcal{C}$ is not definable in order-invariant MSO. However the following formula of MSO(TC) defines $\mathcal{C}$.

$$\exists X \forall Y \forall Z \big(\forall x (X(x)) \wedge (\mathrm{size}(Y,1) \vee \mathrm{size}(Z,1) \vee \neg \times (Y,Z,X))\big) \wedge \exists x \exists y \neg x = y.$$

Corollary

For each vocabulary τ we have that MSO(TC) $\not\leq$ *order-inv* MSO.

Order invariant MSO and MSO(TC)

Order-invariant MSO (on unary vocabularies) is regular languages that are invariant under letter count.

Theorem

Over unary vocabularies MSO(TC) *is strictly more expressive than order-invariant* MSO*.*

Proof.

The proof is based on Parikh's Theorem (1966)

- For every regular language L its letter count is a finite union of linear sets.

□

Order invariant MSO and MSO(TC)

Order-invariant MSO (on unary vocabularies) is regular languages that are invariant under letter count.

Theorem

Over unary vocabularies MSO(TC) *is strictly more expressive than order-invariant* MSO.

Proof.

The proof is based on Parikh's Theorem (1966)

- For every regular language L its letter count is a finite union of linear sets.

□

A subset S of $\mathbb{N}^k$ is a *linear set* if

$$S = \{\vec{v}_0 + \sum_{i=1}^{m} a_i \vec{v}_i \mid a_1, \dots, a_m \in \mathbb{N}\}$$

for some *offset* $\vec{v}_0 \in \mathbb{N}^k$ and *generators* $\vec{v}_1, \dots, \vec{v}_m \in \mathbb{N}^k$.

Open question

- Does the exists an LFP-formula that is not expressible in MSO(TC). On ordered structures, this would show that there are problems in P that are not in NLIN, which is open (it is only know that the two classes are different).
  - EVEN is definable in MSO(TC) but not in LFP.
- What is the relationship of MSO(TC) and order-invariant MSO over vocabularies of higher arity?

Open question

Thanks!

- Does the exists an LFP-formula that is not expressible in MSO(TC). On ordered structures, this would show that there are problems in P that are not in NLIN, which is open (it is only know that the two classes are different).
  - EVEN is definable in MSO(TC) but not in LFP.
- What is the relationship of MSO(TC) and order-invariant MSO over vocabularies of higher arity?