Descriptive Complexity of Deterministic Polylogarithmic Time

Jonni Virtema

Hasselt University, Belgium
jonni.virtema@gmail.com

Joint work with Flavio Ferrarotti, Senén González, José María Turull Torres, and Jan Van den Bussche.

WoLLIC 2019 – July 4th 2019

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

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
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

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
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

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.
  - $\mathrm{ESO}^{\mathrm{polylog}}$ characterises $\mathrm{NPolylogTime}$.
  - Second-order logic characterises the polynomial time hierarchy.
  - Least fixed point logic LFP characterises P on ordered structures.
  - ...
  - Major open problem: Does there exist a logic for P?

Sublinear Complexity Classes and Random Access Machines

- In sublinear time the whole the input cannot be read.
  - Turing machines with sequental access to the input does not suffice.
  - Instead random access model is used (cf. random access memory RAM)
- Random access machine model:

1	0	0	1	1	0	B	$\cdots$	**Input Tape (read only)**
1	0	B	B	B	B	B	$\cdots$	**Address Tape**
0	1	1	0	1	B	B	$\cdots$	**k Work Tapes**

  - Finite control of the machine as for TM.
- PolylogTIME $= \bigcup_{k \in \mathbb{N}} \mathrm{DTIME}[\log^k n]$

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Sublinear Complexity Classes and Random Access Machines

- In sublinear time the whole the input cannot be read.
  - Turing machines with sequental access to the input does not suffice.
  - Instead random access model is used (cf. random access memory RAM)
- Random access machine model:

1	0	0	1	1	0	B	···	**Input Tape (read only)**
1	0	B	B	B	B	B	···	**Address Tape**
0	1	1	0	1	B	B	···	**k Work Tapes**

  - Finite control of the machine as for TM.
- PolylogTIME $= \bigcup_{k \in \mathbb{N}} \mathrm{DTIME}[\log^k n]$

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

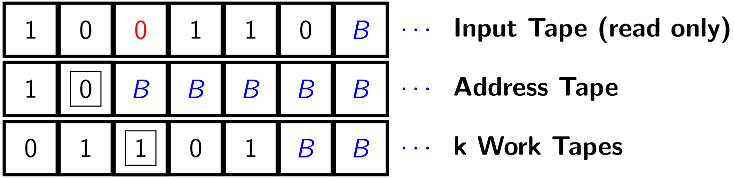
Semantics of IL

Results

Open Question

Sublinear Complexity Classes and Random Access Machines

- In sublinear time the whole the input cannot be read.
  - Turing machines with sequental access to the input does not suffice.
  - Instead random access model is used (cf. random access memory RAM)
- Random access machine model:



  - Finite control of the machine as for TM.
- $\text{PolylogTIME} = \bigcup_{k\in\mathbb{N}} \mathrm{DTIME}[\log^k n]$

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Example computation in deterministic polylogarithmic time

- Calculate the length n of the input.

1	0	0	B	B	B	B	$\cdots$	**Input Tape (read only)**
0	B	B	B	B	B	B	$\cdots$	**Index Tape**

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Example computation in deterministic polylogarithmic time

- Calculate the length n of the input.

1	0	0	B	B	B	B	$\cdots$	**Input Tape (read only)**
1	0	0	B	B	B	B	$\cdots$	**Index Tape**

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Example computation in deterministic polylogarithmic time

- Calculate the length n of the input.

1	0	0	B	B	B	B	$\cdots$	**Input Tape (read only)**
1	0	B	B	B	B	B	$\cdots$	**Index Tape**

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Example computation in deterministic polylogarithmic time

- Calculate the length n of the input.

1	0	0	B	B	B	B	$\cdots$	**Input Tape (read only)**
1	$\boxed{1}$	B	B	B	B	B	$\cdots$	**Index Tape**

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

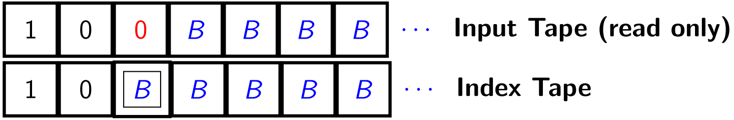
Semantics of IL

Results

Open Question

Example computation in deterministic polylogarithmic time

- Calculate the length n of the input.



  - The index tape has $n - 1$ as binary.
- Any polynomial time numerical property of n (in binary) can be computed.

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Structures as inputs to the Turing machine

- Finite ordered structures with domain $\{0, \ldots, n\}$ and finite vocabularies.
- Structures are encoded as strings as usual in descriptive complexity.
- Relation $R^{\mathbf{A}}$ of arity k is encoded as a binary string of length $|A|^k$, where 1 in a given position indicates that the corresponding tuple is in the relation.
- Constant number $c^{\mathbf{A}}$ is encoded as a binary string of length $\lceil \log n \rceil$.
- k-ary functions are viewed as $\lceil \log n \rceil$-many k-ary relations, where the i-th relation indicates whether the i-th bit is 1.
- $\mathrm{DTIME}[\log^k \hat{n}] = \mathrm{DTIME}[\log^k n]$, where $\hat{n}$ is the length of the encoding and n the domain size.

Structures as inputs to the Turing machine

- Finite ordered structures with domain $\{0, \ldots, n\}$ and finite vocabularies.
- Structures are encoded as strings as usual in descriptive complexity.
- Relation $R^{\mathbf{A}}$ of arity k is encoded as a binary string of length $|A|^k$, where 1 in a given position indicates that the corresponding tuple is in the relation.
- Constant number $c^{\mathbf{A}}$ is encoded as a binary string of length $\lceil \log n \rceil$.
- k-ary functions are viewed as $\lceil \log n \rceil$-many k-ary relations, where the i-th relation indicates whether the i-th bit is 1.
- $\mathrm{DTIME}[\log^k \hat{n}] = \mathrm{DTIME}[\log^k n]$, where $\hat{n}$ is the length of the encoding and n the domain size.

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Structures as inputs to the Turing machine

- Finite ordered structures with domain $\{0, \ldots, n\}$ and finite vocabularies.
- Structures are encoded as strings as usual in descriptive complexity.
- Relation $R^{\mathbf{A}}$ of arity k is encoded as a binary string of length $|A|^k$, where 1 in a given position indicates that the corresponding tuple is in the relation.
- Constant number $c^{\mathbf{A}}$ is encoded as a binary string of length $\lceil \log n \rceil$.
- k-ary functions are viewed as $\lceil \log n \rceil$-many k-ary relations, where the i-th relation indicates whether the i-th bit is 1.
- $\mathrm{DTIME}[\log^k \hat{n}] = \mathrm{DTIME}[\log^k n]$, where $\hat{n}$ is the length of the encoding and n the domain size.

Structures as inputs to the Turing machine

- Finite ordered structures with domain $\{0, \ldots, n\}$ and finite vocabularies.
- Structures are encoded as strings as usual in descriptive complexity.
- Relation $R^{\mathbf{A}}$ of arity k is encoded as a binary string of length $|A|^k$, where 1 in a given position indicates that the corresponding tuple is in the relation.
- Constant number $c^{\mathbf{A}}$ is encoded as a binary string of length $\lceil \log n \rceil$.
- k-ary functions are viewed as $\lceil \log n \rceil$-many k-ary relations, where the i-th relation indicates whether the i-th bit is 1.
- $\mathrm{DTIME}[\log^k \hat{n}] = \mathrm{DTIME}[\log^k n]$, where $\hat{n}$ is the length of the encoding and n the domain size.

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Structures as inputs to the Turing machine

- Finite ordered structures with domain $\{0, \ldots, n\}$ and finite vocabularies.
- Structures are encoded as strings as usual in descriptive complexity.
- Relation $R^{\mathbf{A}}$ of arity k is encoded as a binary string of length $|A|^k$, where 1 in a given position indicates that the corresponding tuple is in the relation.
- Constant number $c^{\mathbf{A}}$ is encoded as a binary string of length $\lceil \log n \rceil$.
- k-ary functions are viewed as $\lceil \log n \rceil$-many k-ary relations, where the i-th relation indicates whether the i-th bit is 1.
- $\mathrm{DTIME}[\log^k \hat{n}] = \mathrm{DTIME}[\log^k n]$, where $\hat{n}$ is the length of the encoding and n the domain size.

Index Logic

- Two sorted structures:
  - Domain of the structure: $\{0, \ldots, n\}$, for some n.
  - Built-in order predicate $\leq$ for the domain.
  - Functions, constants, relations and first-order variables range over the domain.
  - Numerical domain: $\{0, \ldots, \lceil \log n \rceil - 1\}$.
  - Built-in order predicate $\leq$ for the numerical domain.
  - First-order and second-order variables ranging over the numerical domain.
- Vars $x, y, \ldots$ range over the domain, and $\nu, \mu, \ldots$ over the numerical one.
- Idea: Full fixed point logic over the numerical sort, and restricted quantification over the actual domain.

Index Logic

- Two sorted structures:
  - Domain of the structure: $\{0, \ldots, n\}$, for some n.
  - Built-in order predicate $\leq$ for the domain.
  - Functions, constants, relations and first-order variables range over the domain.
  - Numerical domain: $\{0, \ldots, \lceil \log n \rceil - 1\}$.
  - Built-in order predicate $\leq$ for the numerical domain.
  - First-order and second-order variables ranging over the numerical domain.
- Vars $x, y, \ldots$ range over the domain, and $\nu, \mu, \ldots$ over the numerical one.
- Idea: Full fixed point logic over the numerical sort, and restricted quantification over the actual domain.

Fixpoints

Let $F\colon \mathcal{P}(B) \to \mathcal{P}(B)$ be a function.

- X is a fixed point of F, if $F(X) = X$.
- X is the least fixed point, if additionally $X \subseteq Y$ for all other fixed points Y.

For monotonic functions, the least fixed $\mathrm{lfp}(F)$ point always exists.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F(F^m)$$

For non-monotonic functions, we may take the inflationary fixed point $\mathrm{ifp}(F)$.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F^m \cup F(F^m)$$

Fixpoints

Let $F\colon \mathcal{P}(B) \to \mathcal{P}(B)$ be a function.

- X is a fixed point of F, if $F(X) = X$.
- X is the least fixed point, if additionally $X \subseteq Y$ for all other fixed points Y.

For monotonic functions, the least fixed $\mathrm{lfp}(F)$ point always exists.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F(F^m)$$

For non-monotonic functions, we may take the inflationary fixed point $\mathrm{ifp}(F)$.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F^m \cup F(F^m)$$

Fixpoints

Let $F : \mathcal{P}(B) \to \mathcal{P}(B)$ be a function.

- X is a fixed point of F, if $F(X) = X$.
- X is the least fixed point, if additionally $X \subseteq Y$ for all other fixed points Y.

For monotonic functions, the least fixed $\mathrm{lfp}(F)$ point always exists.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F(F^m)$$

For non-monotonic functions, we may take the inflationary fixed point $\mathrm{ifp}(F)$.
It can be calculated as the limit of the process:

$$F^0 = \emptyset, \quad F^{m+1} = F^m \cup F(F^m)$$

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Fixed point logics

- Let $\varphi(X, \bar{x})$ be a formula with a free k-ary relation variable, and $\bar{x}$ a k-tuple of variables.
- On a model $\mathbf{A}, s$, the formula $\varphi(X, \bar{x})$ defines a function $F^{\mathbf{A},s}_{\varphi,X,\bar{x}} \colon \mathcal{P}(A^k) \to \mathcal{P}(A^k)$:

$$F^{\mathbf{A},s}_{\varphi,X,\bar{x}}(B) := \{\bar{a} \mid \mathbf{A}, s(X \mapsto B, \bar{x} \mapsto \bar{a}) \models \varphi\}.$$

- We may take the least fixed point or inflationary fixed point of $F^{\mathbf{A},s}_{\varphi,X,\bar{x}}$.

Fixed point logics

- Let $\varphi(X, \bar{x})$ be a formula with a free k-ary relation variable, and $\bar{x}$ a k-tuple of variables.
- On a model $\mathbf{A}, s$, the formula $\varphi(X, \bar{x})$ defines a function $F^{\mathbf{A},s}_{\varphi,X,\bar{x}} \colon \mathcal{P}(A^k) \to \mathcal{P}(A^k)$:

$$F^{\mathbf{A},s}_{\varphi,X,\bar{x}}(B) := \{\bar{a} \mid \mathbf{A}, s(X \mapsto B, \bar{x} \mapsto \bar{a}) \models \varphi\}.$$

- We may take the least fixed point or inflationary fixed point of $F^{\mathbf{A},s}_{\varphi,X,\bar{x}}$.

Index Logic – Syntax

- Ordinary terms: $t ::= x \mid c \mid f(t, \ldots, t)$.
- Numerical terms: Only numerical variables μ, etc.

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Index Logic – Syntax

- Atomic formulae:

$$\varphi ::= t = t' \mid t \leq t' \mid \mu = \mu' \mid \mu \leq \mu' \mid R(t_1, \ldots, t_n) \mid X(\mu_1, \ldots, \mu_k) \mid$$

Index Logic – Syntax

- Atomic formulae:

$$\varphi ::= t = t' \mid t \leq t' \mid \mu = \mu' \mid \mu \leq \mu' \mid R(t_1, \ldots, t_n) \mid X(\mu_1, \ldots, \mu_k) \mid$$

- More atomic formulae

$$t = \mathit{index}\{\mu : \varphi(\mu)\} \mid [\mathrm{LFP}_{\bar{\mu}, X}\varphi]\bar{\nu} \mid$$

Index Logic – Syntax

- Atomic formulae:

$$\varphi ::= t = t' \mid t \leq t' \mid \mu = \mu' \mid \mu \leq \mu' \mid R(t_1, \ldots, t_n) \mid X(\mu_1, \ldots, \mu_k) \mid$$

- More atomic formulae

$$t = index\{\mu : \varphi(\mu)\} \mid [\mathrm{LFP}_{\bar{\mu}, X}\varphi]\bar{\nu} \mid$$

- Complex formulae

$$\varphi \land \varphi \mid \neg\varphi \mid \exists\mu\varphi \mid \exists x\big(x = index\{\mu : \alpha(\mu)\} \land \varphi\big)$$

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Index Logic – Semantics

$\mathbf{A}, s \models t_1 = t_2$ iff $s(t_1) = s(t_2)$

$\mathbf{A}, s \models t_1 \leq t_2$ iff $s(t_1) \leq s(t_2)$

$\mathbf{A}, s \models R(t_1, \ldots, t_k)$ iff $(s(t_1), \ldots, s(t_k)) \in R^{\mathbf{A}}$

$\mathbf{A}, s \models X(\mu_1, \ldots, \mu_k)$ iff $(s(\mu_1), \ldots, s(\mu_k)) \in s(X)$

$\mathbf{A}, s \models \neg\varphi$ iff $\mathbf{A}, s \not\models \varphi$

$\mathbf{A}, s \models \varphi \wedge \psi$ iff $\mathbf{A}, s \models \varphi$ and $\mathbf{A}, s \models \psi$

$\mathbf{A}, s \models \varphi \vee \psi$ iff $\mathbf{A}, s \models \varphi$ or $\mathbf{A}, s \models \psi$

$\mathbf{A}, s \models \exists\mu\,\varphi$ iff $\mathbf{A}, s(\mu \mapsto i) \models \varphi$, for some $i \leq \lceil \log|A| \rceil$

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Index Logic – Semantics

$\mathbf{A}, s \models t = index\{\mu : \varphi(\mu)\}$ iff

$s(t)$ in binary is $\bar{b}$, where the ith bit is 1 iff $\mathbf{A}, s(\mu \mapsto i) \models \varphi$

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Index Logic – Semantics

$\mathbf{A}, s \models \exists x(x = \mathit{index}\{\mu : \alpha(\mu)\} \land \varphi)$ iff

$\mathbf{A}, s(x \mapsto i) \models x = \mathit{index}\{\mu : \alpha(\mu)\} \land \varphi$, for some $i \in A$.

Index Logic – Semantics

$$\mathbf{A}, s \models [\mathrm{LFP}_{\bar{\mu},X}\varphi]\bar{\nu} \text{ iff } s(\bar{\nu}) \in \mathrm{lfp}(F^{\mathbf{A}}_{\varphi,\bar{\mu},X}).$$

Descriptive Complexity

Polylogarithmic Time

Index Logic

Fixed points

Syntax on IL

Semantics of IL

Results

Open Question

Results

Theorem

Over ordered structures, index logic captures PolylogTime.

Theorem

Let c and d be constant symbols in a vocabulary σ. There does not exist an index logic formula φ that does not use the order predicate $\leq$ on ordinary terms and that is equivalent with the formula $c \leq d$.

Theorem

Let σ be a vocabulary without constant or function symbols. For every sentence φ of index logic of vocabulary σ there exists an equivalent sentence φ' that does not use the order predicate on ordinary terms.

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Proposition

Checking emptiness of a unary relation P^{A} is not computable in $\mathrm{PolylogTime}$*.*
Hence $\exists x P(x)$ is not expressible in index logic.

Proof.

- Let M be a TM that decides in $\mathrm{PolylogTime}$ whether $P^{\mathbf{A}}$ is empty.
  Let f be a polylogarithmic function that bounds the running time of M.
- Let $\mathbf{A}_{\emptyset}$ be the $\{P\}$-structure with domain $\{0, \dots, n-1\}$, where $P^{\mathbf{A}} = \emptyset$.
  The encoding of $\mathbf{A}_{\emptyset}$ to the Turing machine M is the sequence $s := \underbrace{0 \dots 0}_{n \text{ times}}$.
- The running time of M with input s is strictly less than n.
  Let i be an index of s that was not read in the computation $M(s)$.
- Define $s' := \underbrace{0 \dots 0}_{i \text{ times}} 1 \underbrace{0 \dots 0}_{n-i-1 \text{ times}}$.
- The output of the computations $M(s)$ and $M(s')$ are identical.

□

Proposition

Checking emptiness of a unary relation $P^{\mathfrak{A}}$ is not computable in PolylogTime. Hence $\exists x P(x)$ is not expressible in index logic.

Proof.

- Let M be a TM that decides in PolylogTime whether $P^{\mathbf{A}}$ is empty. Let f be a polylogarithmic function that bounds the running time of M.
- Let $\mathbf{A}_\emptyset$ be the $\{P\}$-structure with domain $\{0, \ldots, n-1\}$, where $P^{\mathbf{A}} = \emptyset$. The encoding of $\mathbf{A}_\emptyset$ to the Turing machine M is the sequence $s := \underbrace{0 \ldots 0}_{n \text{ times}}$.
- The running time of M with input s is strictly less than n. Let i be an index of s that was not read in the computation $M(s)$.
- Define $s' := \underbrace{0 \ldots 0}_{i \text{ times}} 1 \underbrace{0 \ldots 0}_{n-i-1 \text{ times}}$.
- The output of the computations $M(s)$ and $M(s')$ are identical.

□

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Proposition

Checking emptiness of a unary relation P^{A} is not computable in $\mathrm{PolylogTime}$. Hence $\exists x P(x)$ is not expressible in index logic.

Proof.

- Let M be a TM that decides in $\mathrm{PolylogTime}$ whether $P^{\mathbf{A}}$ is empty.
  Let f be a polylogarithmic function that bounds the running time of M.
- Let $\mathbf{A}_{\emptyset}$ be the $\{P\}$-structure with domain $\{0, \ldots, n-1\}$, where $P^{\mathbf{A}} = \emptyset$.
  The encoding of $\mathbf{A}_{\emptyset}$ to the Turing machine M is the sequence $s := \underbrace{0 \ldots 0}_{n \text{ times}}$.
- The running time of M with input s is strictly less than n.
  Let i be an index of s that was not read in the computation $M(s)$.
- Define $s' := \underbrace{0 \ldots 0}_{i \text{ times}} 1 \underbrace{0 \ldots 0}_{n-i-1 \text{ times}}$.
- The output of the computations $M(s)$ and $M(s')$ are identical.

Proposition

Checking emptiness of a unary relation P^{A} is not computable in PolylogTime. Hence $\exists x P(x)$ is not expressible in index logic.

Proof.

- Let M be a TM that decides in PolylogTime whether $P^{\mathbf{A}}$ is empty. Let f be a polylogarithmic function that bounds the running time of M.
- Let $\mathbf{A}_\emptyset$ be the $\{P\}$-structure with domain $\{0, \ldots, n-1\}$, where $P^{\mathbf{A}} = \emptyset$. The encoding of $\mathbf{A}_\emptyset$ to the Turing machine M is the sequence $s := \underbrace{0 \ldots 0}_{n \text{ times}}$.
- The running time of M with input s is strictly less than n. Let i be an index of s that was not read in the computation $M(s)$.
- Define $s' := \underbrace{0 \ldots 0}_{i \text{ times}} 1 \underbrace{0 \ldots 0}_{n-i-1 \text{ times}}$.
- The output of the computations $M(s)$ and $M(s')$ are identical.

Proposition

Checking emptiness of a unary relation P^{A} *is not computable in* PolylogTime. *Hence* $\exists x P(x)$ *is not expressible in index logic.*

Proof.

- Let M be a TM that decides in PolylogTime whether $P^{\mathbf{A}}$ is empty.
  Let f be a polylogarithmic function that bounds the running time of M.
- Let $\mathbf{A}_\emptyset$ be the $\{P\}$-structure with domain $\{0, \ldots, n-1\}$, where $P^{\mathbf{A}} = \emptyset$.
  The encoding of $\mathbf{A}_\emptyset$ to the Turing machine M is the sequence $s := \underbrace{0 \ldots 0}_{n \text{ times}}$.
- The running time of M with input s is strictly less than n.
  Let i be an index of s that was not read in the computation $M(s)$.
- Define $s' := \underbrace{0 \ldots 0}_{i \text{ times}} 1 \underbrace{0 \ldots 0}_{n-i-1 \text{ times}}$.
- The output of the computations $M(s)$ and $M(s')$ are identical.

□

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Direct Access Turing Machines

- A novel variant on RAM that accesses the structure directly.
- For each k-ary relation

0	1	1	0	1	B	B	$\cdots$	**k Address Tapes**

- For each k-ary function

0	1	1	0	1	B	B	$\cdots$	**k Address Tapes**
0	1	1	0	1	B	B	$\cdots$	**1 Function Value Tape (Read Only)**

- Additionally

0	1	1	0	1	B	B	$\cdots$	**1 Extra Read Only Tape (stores $\|A\|$)**
0	1	1	0	1	B	B	$\cdots$	**k Work Tapes**

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question

Open Question

- Order-invariant properties are properties of ordered models that remain unaffected if the ordering is redefined.
- Which order-invariant properties are computable in PolylogTime?
- E.g., any polynomial-time numerical property of the size of the domain is clearly computable. For example even cardinality is computable.
- The binary representation of a constant can be computed. However the number depends on the order.

Open Question

Thanks!

- Order-invariant properties are properties of ordered models that remain unaffected if the ordering is redefined.
- Which order-invariant properties are computable in $\mathrm{PolylogTime}$?
- E.g., any polynomial-time numerical property of the size of the domain is clearly computable. For example even cardinality is computable.
- The binary representation of a constant can be computed. However the number depends on the order.

Descriptive Complexity
Polylogarithmic Time
Index Logic
Fixed points
Syntax on IL
Semantics of IL
Results
Open Question