Characterizing the Exponential-Space Hierarchy via Partial Fixpoints

Florian Bruse David Kronenberger Martin Lange
University of Kassel, Germany

FICS workshop
Naples, February 21st, 2024

Descriptive Complexity

characterize complexity classes via logical ressources. $\mathcal{L}$ captures $\mathcal{C}$ if:

1) queries defined by formulas in $\mathcal{L}$ can be computed in $\mathcal{C}$
2) queries that can be computed in $\mathcal{C}$ can be defined in $\mathcal{L}$

only (boolean) queries on finite structures of interest

examples:

- $\exists\mathsf{SO} = \mathsf{NP}$ ([Fagin'74])

Descriptive Complexity

characterize complexity classes via logical ressources. $\mathcal{L}$ captures $\mathcal{C}$ if:

1) queries defined by formulas in $\mathcal{L}$ can be computed in $\mathcal{C}$
2) queries that can be computed in $\mathcal{C}$ can be defined in $\mathcal{L}$

only (boolean) queries on finite structures of interest

examples:

- $\exists\mathsf{SO} = \mathsf{NP}$ ([Fagin'74])
- $\mathsf{FO} + \mathsf{LFP} = \mathsf{PTIME}$ over ordered str. ([Immerman'84], [Vardi'82])

Descriptive Complexity

characterize complexity classes via logical ressources. $\mathcal{L}$ captures $\mathcal{C}$ if:

1) queries defined by formulas in $\mathcal{L}$ can be computed in $\mathcal{C}$
2) queries that can be computed in $\mathcal{C}$ can be defined in $\mathcal{L}$

only (boolean) queries on finite structures of interest

examples:

- $\exists\mathsf{SO} = \mathsf{NP}$ ([Fagin'74])
- $\mathsf{FO} + \mathsf{LFP} = \mathsf{PTIME}$ over ordered str. ([Immerman'84], [Vardi'82])
- $\mathsf{FO} + \mathsf{PFP} = \mathsf{PSPACE}$ ([Vardi'82])

Descriptive Complexity

characterize complexity classes via logical ressources. $\mathcal{L}$ captures $\mathcal{C}$ if:

1) queries defined by formulas in $\mathcal{L}$ can be computed in $\mathcal{C}$
2) queries that can be computed in $\mathcal{C}$ can be defined in $\mathcal{L}$

only (boolean) queries on finite structures of interest

examples:

- $\exists\mathsf{SO} = \mathsf{NP}$ ([Fagin'74])
- $\mathsf{FO} + \mathsf{LFP} = \mathsf{PTIME}$ over ordered str. ([Immerman'84], [Vardi'82])
- $\mathsf{FO} + \mathsf{PFP} = \mathsf{PSPACE}$ ([Vardi'82])

Some extensions/generalizations:

- $\mathsf{SO} = \mathsf{PH}$ ([Stockmeyer'76])
- $\mathsf{HO}^{k+1} + \mathsf{LFP} = k{-}\mathsf{EXPTIME}$ over ordered* str. ([Freire/Martins'11])

Descriptive Complexity

characterize complexity classes via logical ressources. $\mathcal{L}$ captures $\mathcal{C}$ if:

1) queries defined by formulas in $\mathcal{L}$ can be computed in $\mathcal{C}$
2) queries that can be computed in $\mathcal{C}$ can be defined in $\mathcal{L}$

only (boolean) queries on finite structures of interest

examples:

- $\exists\mathrm{SO} = \mathrm{NP}$ ([Fagin'74])
- $\mathrm{FO} + \mathrm{LFP} = \mathrm{PTIME}$ over ordered str. ([Immerman'84], [Vardi'82])
- $\mathrm{FO} + \mathrm{PFP} = \mathrm{PSPACE}$ ([Vardi'82])

Some extensions/generalizations:

- $\mathrm{SO} = \mathrm{PH}$ ([Stockmeyer'76])
- $\mathrm{HO}^{k+1} + \mathrm{LFP} = k\mathrm{-EXPTIME}$ over ordered* str. ([Freire/Martins'11])
- $\mathrm{HO}^{k+1} + \mathrm{PFP} = k\mathrm{-EXPSPACE}$ (this talk)

$*$: for $k = 0$

Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f\colon X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \vee \exists y.\ E(x,y) \wedge X(y)$ (reachability of a node where p holds)

Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f : X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathsf{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathsf{LFP}\ X.\ p(x) \lor \exists y.\ E(x,y) \land X(y)$ (reachability of a node where p holds)

Ex.: $\mathsf{LFP}\ X.\ \big(X(x) \land \exists y.\ x > y \land \neg X(y)\big) \lor \big(\neg X(x) \land \forall y.\ x > y \to X(y)\big)$

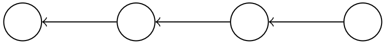
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f : X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \lor \exists y.\ E(x,y) \land X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ \big(X(x) \land \exists y.\ x > y \land \neg X(y)\big) \lor \big(\neg X(x) \land \forall y.\ x > y \to X(y)\big)$



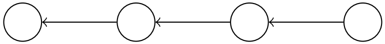
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f : X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \lor \exists y.\ E(x,y) \land X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ \big(X(x) \land \exists y.\ x > y \land \neg X(y)\big) \lor \big(\neg X(x) \land \forall y.\ x > y \to X(y)\big)$



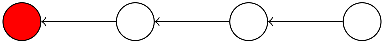
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f : X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \vee \exists y.\ E(x,y) \wedge X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ \big(X(x) \wedge \exists y.\ x > y \wedge \neg X(y)\big) \vee \big(\neg X(x) \wedge \forall y.\ x > y \rightarrow X(y)\big)$



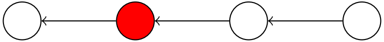
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f\colon X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathsf{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathsf{LFP}\ X.\ p(x) \lor \exists y.\ E(x,y) \land X(y)$ (reachability of a node where p holds)

Ex.: $\mathsf{LFP}\ X.\ \big(X(x) \land \exists y.\ x > y \land \neg X(y)\big) \lor \big(\neg X(x) \land \forall y.\ x > y \to X(y)\big)$



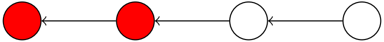
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f: X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \lor \exists y.\ E(x,y) \land X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ \big(X(x) \land \exists y.\ x > y \land \neg X(y)\big) \lor \big(\neg X(x) \land \forall y.\ x > y \to X(y)\big)$



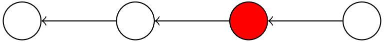
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f\colon X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \vee \exists y.\ E(x,y) \wedge X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ \big(X(x) \wedge \exists y.\ x > y \wedge \neg X(y)\big) \vee \big(\neg X(x) \wedge \forall y.\ x > y \to X(y)\big)$



binary incrementation not monotone $\to$ need partial fixpoint:

- $f\colon X \mapsto f(X)$ yields sequence $\emptyset, f(\emptyset), f^2(\emptyset), \ldots$
- $\mathrm{PFP}\ f = \begin{cases} f^i(\emptyset), & \text{if } f^i(\emptyset) = f^{i+1}(\emptyset) \\ \emptyset, & \text{otherwise} \end{cases}$

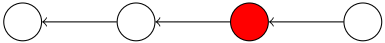
Partial Fixpoints

Knaster-Tarski/Kleene guarantee well-definedness of fixpoints of monotone functions:

- $f : X \mapsto f(X)$ (in powerset lattice) yields sequence $\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \cdots$
- least fixpoint $\mathrm{LFP}\ f$ defined as first stable element of sequence

Ex.: $\mathrm{LFP}\ X.\ p(x) \vee \exists y.\ E(x,y) \wedge X(y)$ (reachability of a node where p holds)

Ex.: $\mathrm{LFP}\ X.\ (X(x) \wedge \exists y.\ x > y \wedge \neg X(y)) \vee (\neg X(x) \wedge \forall y.\ x > y \rightarrow X(y))$



binary incrementation not monotone $\rightarrow$ need partial fixpoint:

- $f : X \mapsto f(X)$ yields sequence $\emptyset, f(\emptyset), f^2(\emptyset), \ldots$
- $\mathrm{PFP}\ f = \begin{cases} f^i(\emptyset), & \text{if } f^i(\emptyset) = f^{i+1}(\emptyset) \\ \emptyset, & \text{otherwise} \end{cases}$

Obs.: sequence either stabilizes after at most exponentially many steps, or it cycles

Higher-Order Logic (+ PFP)

Types:

$$\tau ::= \bullet \text{ (individuals)} \mid \tau_1, \ldots, \tau_n \text{ (cross product)} \mid (\tau) \text{ sets}$$

- $\mathrm{ord}(\bullet) = 1$,
- $\mathrm{ord}(\tau_1, \ldots, \tau_n) = \max\{\mathrm{ord}(\tau_1), \ldots, \mathrm{ord}(\tau_n)\}$
- $\mathrm{ord}((\tau)) = 1 + \mathrm{ord}(\tau)$

Higher-Order Logic (+ PFP)

Types:

$$\tau ::= \bullet \text{ (individuals)} \mid \tau_1, \dots, \tau_n \text{ (cross product)} \mid (\tau) \text{ sets}$$

- $\mathrm{ord}(\bullet) = 1$,
- $\mathrm{ord}(\tau_1, \dots, \tau_n) = \max\{\mathrm{ord}(\tau_1), \dots, \mathrm{ord}(\tau_n)\}$
- $\mathrm{ord}((\tau)) = 1 + \mathrm{ord}(\tau)$

Formulas of HO :

$$\varphi ::= p(X) \mid E(X, Y) \mid X(\vec{Y}) \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists X^{\tau}.\ \varphi$$

where $\vec{Y} = Y_1, \dots, Y_n$, type annotations dropped where possible

Higher-Order Logic (+ PFP)

Types:

$$\tau ::= \bullet \text{ (individuals)} \mid \tau_1, \ldots, \tau_n \text{ (cross product)} \mid (\tau) \text{ sets}$$

- $\mathrm{ord}(\bullet) = 1$,
- $\mathrm{ord}(\tau_1, \ldots, \tau_n) = \max\{\mathrm{ord}(\tau_1), \ldots, \mathrm{ord}(\tau_n)\}$
- $\mathrm{ord}((\tau)) = 1 + \mathrm{ord}(\tau)$

Formulas of HO :

$$\varphi ::= p(X) \mid E(X, Y) \mid X(\vec{Y}) \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists X^\tau.\, \varphi$$

where $\vec{Y} = Y_1, \ldots, Y_n$, type annotations dropped where possible

Ex.: $<$ (written inline) is strict total order in ψ

$$\exists <^{(\bullet,\bullet)} .\, \psi(<) \wedge \forall x^\bullet, y^\bullet, z^\bullet.\, (x < y \wedge y < z \rightarrow x < z)$$
$$\wedge\, x \not< x \wedge (x \neq y \rightarrow x < y \vee y < x)$$

Obs.: w.l.o.g. structures ordered from now on

Higher-Order Logic (+ PFP)

Types:

$$\tau ::= \bullet \text{ (individuals)} \mid \tau_1, \ldots, \tau_n \text{ (cross product)} \mid (\tau) \text{ sets}$$

- $\mathrm{ord}(\bullet) = 1$,
- $\mathrm{ord}(\tau_1, \ldots, \tau_n) = \max\{\mathrm{ord}(\tau_1), \ldots, \mathrm{ord}(\tau_n)\}$
- $\mathrm{ord}((\tau)) = 1 + \mathrm{ord}(\tau)$

Formulas of HO+PFP:

$$\varphi ::= p(X) \mid E(X, Y) \mid X(\vec{Y}) \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists X^\tau.\, \varphi \mid (\mathrm{PFP}\ X^\tau.\, \varphi)(\vec{Y})$$

where $\vec{Y} = Y_1, \ldots, Y_n$, type annotations dropped where possible

Ex.: $<$ (written inline) is strict total order in ψ

$$\exists <^{(\bullet,\bullet)} .\ \psi(<) \wedge \forall x^\bullet, y^\bullet, z^\bullet.\ (x < y \wedge y < z \to \ x < z) \wedge x \not< x \wedge (x \neq y \to x < y \vee y < x)$$

Obs.: w.l.o.g. structures ordered from now on

Higher-Order Logic (+ PFP), cont'd.

Ex.: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \land \varphi_{\mathsf{init}}(x)) \lor (\exists y^{\tau}.\varphi_{\mathsf{step}}(x, y) \land X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Higher-Order Logic (+ PFP), cont'd.

Ex.: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \land \varphi_{\mathrm{init}}(x)) \lor (\exists y^{\tau}.\varphi_{\mathrm{step}}(x, y) \land X(y))\Big)\ \varphi_{\mathrm{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

Higher-Order Logic (+ PFP), cont'd.

Ex.: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \land \varphi_{\mathsf{init}}(x)) \lor (\exists y^{\tau}.\varphi_{\mathsf{step}}(x, y) \land X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

An application:

Ex.:
$\mathrm{PFP}\ X.\ X = Y \lor ((X(x) \land \exists y.\ x > y \land \neg X(y)) \lor (\neg X(x) \land \forall y.\ x > y \rightarrow X(y)))(Z)$

Higher-Order Logic (+ PFP), cont'd.

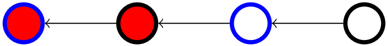
Ex.: $\big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \land \varphi_{\mathsf{init}}(x)) \lor (\exists y^\tau.\varphi_{\mathsf{step}}(x,y) \land X(y))\big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\to$ less space used

An application:

Ex.:
$\mathrm{PFP}\ X.\ X = Y \lor ((X(x) \land \exists y.\ x > y \land \neg X(y)) \lor (\neg X(x) \land \forall y.\ x > y \to X(y)))(Z)$



Higher-Order Logic (+ PFP), cont'd.

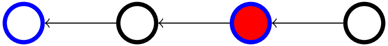
Ex.: $\Big(\mathsf{PFP}\ X^{(\tau)}.\ (X = \emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \wedge X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

An application:

Ex.:
$\mathsf{PFP}\ X.\ X = Y \vee ((X(x) \wedge \exists y.\ x > y \wedge \neg X(y)) \vee (\neg X(x) \wedge \forall y.\ x > y \rightarrow X(y)))(Z)$



Higher-Order Logic (+ PFP), cont'd.

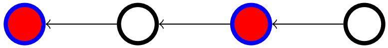
Ex.: $\Big(\mathsf{PFP}\ X^{(\tau)}.\ (X = \emptyset \land \varphi_{\mathsf{init}}(x)) \lor (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \land X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

An application:

Ex.:
$\mathsf{PFP}\ X.\ X = Y \lor ((X(x) \land \exists y.\ x > y \land \neg X(y)) \lor (\neg X(x) \land \forall y.\ x > y \to X(y)))(Z)$



Higher-Order Logic (+ PFP), cont'd.

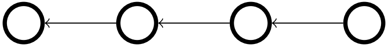
Ex.: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x, y) \wedge X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\to$ less space used

An application:

Ex.:
$\mathrm{PFP}\ X.\ X = Y \vee ((X(x) \wedge \exists y.\ x > y \wedge \neg X(y)) \vee (\neg X(x) \wedge \forall y.\ x > y \to X(y)))(Z)$



Higher-Order Logic (+ PFP), cont'd.

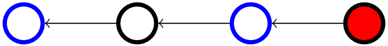
Ex.: $\Big(\mathsf{PFP}\ X^{(\tau)}.\ (X = \emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \wedge X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

An application:

Ex.:
$\mathsf{PFP}\ X.\ X = Y \vee ((X(x) \wedge \exists y.\ x > y \wedge \neg X(y)) \vee (\neg X(x) \wedge \forall y.\ x > y \rightarrow X(y)))(Z)$



Higher-Order Logic (+ PFP), cont'd.

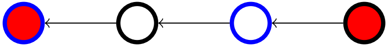
Ex.: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ (X = \emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \wedge X(y))\Big)\ \varphi_{\mathsf{end}}$

with τ as type of x (reachability of some end position from start)

Obs.: this form of reachability test does not store the full set of reachable positions (cf. LFP approach) $\rightarrow$ less space used

An application:

Ex.:
$\mathrm{PFP}\ X.\ X = Y \vee ((X(x) \wedge \exists y.\ x > y \wedge \neg X(y)) \vee (\neg X(x) \wedge \forall y.\ x > y \rightarrow X(y)))(Z)$



Ordering Higher-Order Relations

technical requirement: In order to encode Turing machine runs, we need to count up to large numbers

have already seen that w.l.o.g. all structures ordered

Ordering Higher-Order Relations

technical requirement: In order to encode Turing machine runs, we need to count up to large numbers

have already seen that w.l.o.g. all structures ordered

Ex.: $\varphi_<^{\bullet,\dots,\bullet}(X_1,\dots,X_n,Y_1,\dots,Y_n) = \bigvee_{i=1}^{n} <(X_i,Y_i) \wedge \bigwedge_{j=1}^{i-1} \neg <(Y_j,X_j)$
(first tuple is lexicographically smaller than second one)

Ordering Higher-Order Relations

technical requirement: In order to encode Turing machine runs, we need to count up to large numbers

have already seen that w.l.o.g. all structures ordered

Ex.: $\varphi_<^{\bullet,\dots,\bullet}(X_1,\dots,X_n,Y_1,\dots,Y_n) = \bigvee_{i=1}^{n} < (X_i,Y_i) \land \bigwedge_{j=1}^{i-1} \neg < (Y_j,X_j)$
(first tuple is lexicographically smaller than second one)

Ex.: $\varphi_<^{(\bullet,\dots,\bullet)}(X^{\bullet,\dots,\bullet},Y^{\bullet,\dots,\bullet}) =$
$\exists(\vec{Z}^{\bullet,\dots,\bullet}).\ Y(\vec{Z}) \land \neg X(\vec{Z}) \land \forall(\vec{Z'}^{\bullet,\dots,\bullet}).\ \varphi_<^{\bullet,\dots,\bullet}(\vec{Z},\vec{Z'}) \to (X(\vec{Z'}) \to Y(\vec{Z'}))$
(the first relation is lexicographically smaller than second one)

Ordering Higher-Order Relations

technical requirement: In order to encode Turing machine runs, we need to count up to large numbers

have already seen that w.l.o.g. all structures ordered

Ex.: $\varphi_<^{\bullet,\dots,\bullet}(X_1,\dots,X_n,Y_1,\dots,Y_n) = \bigvee_{i=1}^{n} <(X_i,Y_i) \wedge \bigwedge_{j=1}^{i-1} \neg <(Y_j,X_j)$
(first tuple is lexicographically smaller than second one)

Ex.: $\varphi_<^{(\bullet,\dots,\bullet)}(X^{\bullet,\dots,\bullet},Y^{\bullet,\dots,\bullet}) =$
$\exists(\vec{Z}^{\bullet,\dots,\bullet}).\ Y(\vec{Z}) \wedge \neg X(\vec{Z}) \wedge \forall(\vec{Z'}^{\bullet,\dots,\bullet}).\ \varphi_<^{\bullet,\dots,\bullet}(\vec{Z},\vec{Z'}) \rightarrow (X(\vec{Z'}) \rightarrow Y(\vec{Z'}))$
(the first relation is lexicographically smaller than second one)

can continue this:
Ex.: $\varphi_<^{((\bullet,\dots,\bullet))}(X^{((\bullet,\dots,\bullet))},Y^{((\bullet,\dots,\bullet))}) =$
$\exists Z^{(\bullet,\dots,\bullet)}.\ Y(Z) \wedge \neg X(Z) \wedge \forall Z'^{(\bullet,\dots,\bullet)}.\ \varphi_<^{(\bullet,\dots,\bullet)}(Z',Z) \rightarrow (X(Z') \rightarrow Y(Z'))$

Lemma: All types can be considered ordered from now on

also have formulas to find e.g., first, last element of order

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for $\mathsf{HO}^{k+1} + \mathsf{PFP} = k$-EXPSPACE we need:

1) queries def. by formulas in $\mathsf{HO}^{k+1} + \mathsf{PFP}$ can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in $\mathsf{HO}^{k+1} + \mathsf{PFP}$

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for $\mathrm{HO}^{k+1} + \mathrm{PFP} = k$-EXPSPACE we need:

1) queries def. by formulas in $\mathrm{HO}^{k+1} + \mathrm{PFP}$ can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in $\mathrm{HO}^{k+1} + \mathrm{PFP}$

1) is straightforward

- HO^{k+1} has k-EXPTIME model checking
- PFP of order $k+2$ have only $k+1$-fold exp. many different possible values

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for $\mathrm{HO}^{k+1} + \mathrm{PFP} = k$-EXPSPACE we need:

1) queries def. by formulas in $\mathrm{HO}^{k+1} + \mathrm{PFP}$ can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in $\mathrm{HO}^{k+1} + \mathrm{PFP}$

1) is straightforward

- HO^{k+1} has k-EXPTIME model checking
- PFP of order $k+2$ have only $k+1$-fold exp. many different possible values
  $\rightarrow$ iterate until stable, but stop if ($k+1$-fold exp.) counter runs out (log. coding)

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for HO^{k+1} + PFP = k-EXPSPACE we need:

1) queries def. by formulas in HO^{k+1} + PFP can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in HO^{k+1} + PFP

1) is straightforward

- HO^{k+1} has k-EXPTIME model checking
- PFP of order $k+2$ have only $k+1$-fold exp. many different possible values
  $\rightarrow$ iterate until stable, but stop if ($k+1$-fold exp.) counter runs out (log. coding)

for 2), let

- $\mathcal{M}$ an $\mathrm{EXP}_k^{p(n)}$-space-bounded DTM (p poly.) and $w \in \Sigma^*$ input
- $\mathcal{T}$ labelled graph

Idea:

- encode configurations of $\mathcal{M}$ on input w as (sets of) tuples in $\mathcal{T}$

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for HO^{k+1} + PFP = k-EXPSPACE we need:

1) queries def. by formulas in HO^{k+1} + PFP can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in HO^{k+1} + PFP

1) is straightforward

- HO^{k+1} has k-EXPTIME model checking
- PFP of order $k+2$ have only $k+1$-fold exp. many different possible values
  $\rightarrow$ iterate until stable, but stop if ($k+1$-fold exp.) counter runs out (log. coding)

for 2), let

- $\mathcal{M}$ an $\mathrm{EXP}_k^{p(n)}$-space-bounded DTM (p poly.) and $w \in \Sigma^*$ input
- $\mathcal{T}$ labelled graph

Idea:

- encode configurations of $\mathcal{M}$ on input w as (sets of) tuples in $\mathcal{T}$
- find formula that maps conf. to unique successor (or same conf. if accepting)
- iteration of PFP of that formula simulates run of $\mathcal{M}$ on w

Characterizing k-EXPSPACE

HO^{k+1} = formulas with types of order at most $k+1$ (PFP of order at most $k+2$)

for HO^{k+1} + PFP = k-EXPSPACE we need:

1) queries def. by formulas in HO^{k+1} + PFP can be computed in k-EXPSPACE
2) queries that can be computed in k-EXPSPACE can be def. in HO^{k+1} + PFP

1) is straightforward

- HO^{k+1} has k-EXPTIME model checking
- PFP of order $k+2$ have only $k+1$-fold exp. many different possible values
  $\rightarrow$ iterate until stable, but stop if ($k+1$-fold exp.) counter runs out (log. coding)

for 2), let

- $\mathcal{M}$ an $\mathrm{EXP}_k^{p(n)}$-space-bounded DTM (p poly.) and $w \in \Sigma^*$ input
- $\mathcal{T}$ labelled graph

Idea:

- encode configurations of $\mathcal{M}$ on input w as (sets of) tuples in $\mathcal{T}$
- find formula that maps conf. to unique successor (or same conf. if accepting)
- iteration of PFP of that formula simulates run of $\mathcal{M}$ on w
- query result for (encoding of) accepting conf.

Encoding Configurations of Space-Bounded DTM

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathsf{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \ldots, \mathsf{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

Encoding Configurations of Space-Bounded DTM

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathsf{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \dots, \mathsf{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

domain of t of size $\mathsf{EXP}_k^{p(n)}$ (!)
$\to$ must be careful to model t as one relation together with q, h

Encoding Configurations of Space-Bounded DTM

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathsf{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \dots, \mathsf{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

domain of t of size $\mathsf{EXP}_k^{p(n)}$ (!)
$\to$ must be careful to model t as one relation together with q, h

sorting out the various orders:

- type (τ) with k-fold exponentially many elements must have order $k+2$ (cf. poly $= \mathsf{EXP}_0^{p(n)}$ needs sets of individuals, i.e., order 2)

Encoding Configurations of Space-Bounded DTM

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathrm{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \dots, \mathrm{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

domain of t of size $\mathrm{EXP}_k^{p(n)}$ (!)
$\to$ must be careful to model t as one relation together with q, h

sorting out the various orders:

- type (τ) with k-fold exponentially many elements must have order $k+2$ (cf. poly $= \mathrm{EXP}_0^{p(n)}$ needs sets of individuals, i.e., order 2)
- function with such a domain is (functional) relation between domain and range
- seems to match, but we must iterate on encoding of a configuration

Encoding Configurations of Space-Bounded DTM

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathrm{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \ldots, \mathrm{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

domain of t of size $\mathrm{EXP}_k^{p(n)}$ (!)
$\to$ must be careful to model t as one relation together with q, h

sorting out the various orders:

- type (τ) with k-fold exponentially many elements must have order $k+2$ (cf. poly $= \mathrm{EXP}_0^{p(n)}$ needs sets of individuals, i.e., order 2)
- function with such a domain is (functional) relation between domain and range
- seems to match, but we must iterate on encoding of a configuration $\to$ encoding needs to fit into one variable, incl. state, head position
- yields variable of type $Q \times \tau \times (\tau \times \Gamma) \to$ order $k+2$

Encoding Configurations of Space-Bounded DTM, cont'd.

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathsf{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \dots, \mathsf{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents (Γ = tape alphabet)

represent $\mathcal{C}$ as tuple of the form (s, H, T) where

- $s \in \mathcal{T}$ encodes the state

Encoding Configurations of Space-Bounded DTM, cont'd.

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathsf{EXP}_k^{p(|w|)}$ head position,
- $t \colon \{0, \ldots, \mathsf{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents ($\Gamma =$ tape alphabet)

represent $\mathcal{C}$ as tuple of the form (s, H, T) where

- $s \in \mathcal{T}$ encodes the state
- $H \in [\![\tau]\!]$ encodes head position as number between 0 and $\mathsf{EXP}_k^{p(n)}$

Encoding Configurations of Space-Bounded DTM, cont'd.

configuration of $\mathcal{M}$ on input w is (q, h, t) with

- q state of $\mathcal{M}$
- $0 \leq h \leq \mathrm{EXP}_k^{p(|w|)}$ head position,
- $t\colon \{0, \ldots, \mathrm{EXP}_k^{p(|w|)}\} \to \Gamma$ tape contents ($\Gamma =$ tape alphabet)

represent $\mathcal{C}$ as tuple of the form (s, H, T) where

- $s \in \mathcal{T}$ encodes the state
- $H \in [\![\tau]\!]$ encodes head position as number between 0 and $\mathrm{EXP}_k^{p(n)}$
- $I \in ([\![\tau]\!] \times \mathcal{T})$ encodes tape
- ex. exactly one s' with $(i, s') \in I$ f.a. $i \in [\![\tau]\!]$

NB: A bit more challenging if cross product type not available standalone

Encoding Runs

type (τ) from last slide can be chosen of order $k+2$ (form: $((\cdots(\bullet,\ldots,\bullet)\cdots)))$

- hence a tuple of the form (s, H, T) is also of order $k+2$
  $\rightarrow$ encoding of conf. is object of order $k+2$

Encoding Runs

type (τ) from last slide can be chosen of order $k+2$ (form: $((\cdots(\bullet,\ldots,\bullet)\cdots)))$

- hence a tuple of the form (s, H, T) is also of order $k+2$
  $\rightarrow$ encoding of conf. is object of order $k+2$
- initial and final configurations definable in $\mathrm{HO}^{k+1}{+}\mathrm{PFP}$

Encoding Runs

type (τ) from last slide can be chosen of order $k+2$ (form: $((\cdots(\bullet,\ldots,\bullet)\cdots))$)

- hence a tuple of the form (s, H, T) is also of order $k+2$
  $\rightarrow$ encoding of conf. is object of order $k+2$
- initial and final configurations definable in HO^{k+1}+PFP

Recall: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ X = (\emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \wedge X(y))\Big)\varphi_{\mathsf{end}}$

- formalizing that (encoding of) one conf. is successor of another is possible
- not shown here since transition table of DTM must be encoded (lots of cases)

Encoding Runs

type (τ) from last slide can be chosen of order $k+2$ (form: $((\cdots(\bullet,\ldots,\bullet)\cdots))$)

- hence a tuple of the form (s, H, T) is also of order $k+2$
  $\rightarrow$ encoding of conf. is object of order $k+2$
- initial and final configurations definable in $\mathrm{HO}^{k+1}{+}\mathrm{PFP}$

Recall: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ X = (\emptyset \wedge \varphi_{\mathrm{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathrm{step}}(x,y) \wedge X(y))\Big)\varphi_{\mathrm{acc}}$

- formalizing that (encoding of) one conf. is successor of another is possible
- not shown here since transition table of DTM must be encoded (lots of cases)

Encoding Runs

type (τ) from last slide can be chosen of order $k+2$ (form: $((\cdots(\bullet,\ldots,\bullet)\cdots))$)

- hence a tuple of the form (s, H, T) is also of order $k+2$
  $\rightarrow$ encoding of conf. is object of order $k+2$
- initial and final configurations definable in $\mathrm{HO}^{k+1}{+}\mathrm{PFP}$

Recall: $\Big(\mathrm{PFP}\ X^{(\tau)}.\ X = (\emptyset \wedge \varphi_{\mathsf{init}}(x)) \vee (\exists y^{\tau}.\varphi_{\mathsf{step}}(x,y) \wedge X(y))\Big)\varphi_{\mathsf{acc}}$

- formalizing that (encoding of) one conf. is successor of another is possible
- not shown here since transition table of DTM must be encoded (lots of cases)
- requires some basic arithmetic (but no uniform notion of addition)

Theorem 1

$\mathrm{HO}^{k+1}{+}\mathrm{PFP}$ captures k-EXPSPACE for $k \geq 0$.

Preview: Extending Otto's Theorem

Thm. ([Otto'99]): bisimulation-invariant PTIME captured by the polyadic *mu*-calculus

- bisimulation-invariance: queries must answer the same for bisimilar inputs

Preview: Extending Otto's Theorem

Thm. ([Otto'99]): bisimulation-invariant PTIME captured by the polyadic *mu*-calculus

- bisimulation-invariance: queries must answer the same for bisimilar inputs
- proof uses Immerman-Vardi (PTIME = FO+LFP on ord. str.), then translates between FO+LFP and poly. μ-calc.
- Note: $\sim$ invariance allows us to drop order requirement (since definable)!

Preview: Extending Otto's Theorem

Thm. ([Otto'99]): bisimulation-invariant PTIME captured by the polyadic *mu*-calculus

- bisimulation-invariance: queries must answer the same for bisimilar inputs
- proof uses Immerman-Vardi (PTIME = FO+LFP on ord. str.), then translates between FO+LFP and poly. μ-calc.
- Note: $\sim$ invariance allows us to drop order requirement (since definable)!

Extensions:

Thm. ([B./Kronenberger/Lange'22]): $\sim$-inv. k-EXPTIME is capt. by order-k poly. HFL

- HFL = higher-order modal fixpoint logic ([Viswanathan/Viswanathan'04] obtained by adding simply-typed λ-calculus to μ-calculus, + (function) fixpoints)
- NB: different notion of order
- follows Otto's pattern, use HO^{k+1}+LFP = k-EXPTIME (Freire/Martens'11)

Preview: Extending Otto's Theorem

Thm. ([Otto'99]): bisimulation-invariant PTIME captured by the polyadic *mu*-calculus

- bisimulation-invariance: queries must answer the same for bisimilar inputs
- proof uses Immerman-Vardi (PTIME = FO+LFP on ord. str.), then translates between FO+LFP and poly. μ-calc.
- Note: $\sim$ invariance allows us to drop order requirement (since definable)!

Extensions:

Thm. ([B./Kronenberger/Lange'22]): $\sim$-inv. k-EXPTIME is capt. by order-k poly. HFL

- HFL = higher-order modal fixpoint logic ([Viswanathan/Viswanathan'04] obtained by adding simply-typed λ-calculus to μ-calculus, + (function) fixpoints)
- NB: different notion of order
- follows Otto's pattern, use HO^{k+1}+LFP = k-EXPTIME (Freire/Martens'11)

Thm.: $\sim$-inv. k-EXPSPACE is captured by tail-recursive order-$k+1$ poly. HFL

tail recursion limits interplay between fixpoints and (higher-order) functions

Preview: Extending Otto's Theorem

Thm. ([Otto'99]): bisimulation-invariant PTIME captured by the polyadic *mu*-calculus

- bisimulation-invariance: queries must answer the same for bisimilar inputs
- proof uses Immerman-Vardi (PTIME = FO+LFP on ord. str.), then translates between FO+LFP and poly. μ-calc.
- Note: $\sim$ invariance allows us to drop order requirement (since definable)!

Extensions:

Thm. ([B./Kronenberger/Lange'22]): $\sim$-inv. k-EXPTIME is capt. by order-k poly. HFL

- HFL = higher-order modal fixpoint logic ([Viswanathan/Viswanathan'04] obtained by adding simply-typed λ-calculus to μ-calculus, + (function) fixpoints)
- NB: different notion of order
- follows Otto's pattern, use HO^{k+1}+LFP = k-EXPTIME (Freire/Martens'11)

Thm.: $\sim$-inv. k-EXPSPACE is captured by tail-recursive order-$k+1$ poly. HFL

tail recursion limits interplay between fixpoints and (higher-order) functions

Questions?