I decided to start with a survey of Descriptive Complexity, and end with a survey of Nested Words.

Please ask questions during my talk because two-way communication is more fun and allows much more understanding!
Descriptive Complexity
Descriptive Complexity

Input $q_1 q_2 \cdots q_n$ \hspace{1cm} Computation \hspace{1cm} Output $a_1 a_2 \cdots a_i \cdots a_{n^k}$
Descriptive Complexity

Input
$q_1 \ q_2 \ \cdots \ q_n$

$\mapsto$

Computation

$\mapsto$

Output
$a_1 \ a_2 \ \cdots \ [a_i] \ \cdots \ a_{n^k}$
Descriptive Complexity

Input
$q_1 \ q_2 \ \cdots \ q_n$

$\rightarrow$ Computation

$\rightarrow$ Output
$a$
How much computation is needed to check if input has property $S$?
How much computation is needed to check if input has property $S$?

How rich a language do we need to express property $S$?
Encode Input as Finite Logical Structure

Graph

\[ G = (\{v_1, \ldots, v_n\}, E, s, t) \]

Binary

\[ A_w = (\{p_1, \ldots, p_8\}, S) \]

String

\[ S = \{p_2, p_5, p_7, p_8\} \]

\[ w = 01001011 \]

Vocabularies:

\[ \tau_g = (E^2, s, t), \quad \tau_s = (S^1) \]
First-Order Logic

input symbols: from \( \tau \)
variables: \( x, y, z, \ldots \)
boolean connectives: \( \land, \lor, \neg \)
quantifiers: \( \forall, \exists \)
numeric symbols: \( =, \leq, +, \times, 0, \text{max} \)

\[
\alpha \equiv \forall x \exists y (E(x, y)) \quad \in \quad \mathcal{L}(\tau_g)
\]

\[
\beta \equiv \exists x \forall y (x \leq y \land S(x)) \quad \in \quad \mathcal{L}(\tau_s)
\]

\[
\beta \equiv S(0) \quad \in \quad \mathcal{L}(\tau_s)
\]
Second-Order Logic

= first-order logic + quantifiable relational variables

\[ \Phi_{3\text{-color}} \equiv \exists R^1 Y^1 B^1 \forall x \, y \left( (R(x) \lor Y(x) \lor B(x)) \land (E(x, y) \rightarrow (\neg(R(x) \land R(y)) \land \neg(Y(x) \land Y(y))) \land \neg(B(x) \land B(y)))) \right) \]
Second-Order Logic

Fagin’s Theorem: \[ \text{NP} = \text{SO} \exists \]

\[ \Phi_{3\text{-color}} \equiv \exists R^1 Y^1 B^1 \forall x y ((R(x) \lor Y(x) \lor B(x)) \land \\
(E(x, y) \rightarrow (\neg (R(x) \land R(y)) \land \neg (Y(x) \land Y(y))) \land \\
\quad \land \neg (B(x) \land B(y)))) ) \]
Addition is First-Order

\[ Q_+ : \text{STRUC}[\tau_{AB}] \rightarrow \text{STRUC}[\tau_s] \]

\[
\begin{array}{cccccccc}
A & a_1 & a_2 & \ldots & a_{n-1} & a_n \\
B & + & b_1 & b_2 & \ldots & b_{n-1} & b_n \\
S & \hline & s_1 & s_2 & \ldots & s_{n-1} & s_n \\
\end{array}
\]

\[
C(i) \equiv (\exists j > i) \left( A(j) \land B(j) \land (\forall k. j > k > i)(A(k) \lor B(k)) \right)
\]

\[
Q_+(i) \equiv A(i) \oplus B(i) \oplus C(i)
\]

Descriptive Complexity and Nested Words – 5/46
Parallel Machines

CRAM\[t(n)\] = CRCW-PRAM-TIME\[t(n)\]-HARD\[n^{O(1)}\]

synchronous, concurrent read and write
\(n^{O(1)}\) processors and memory
CRAM[\( t(n) \)] = CRCW-PRAM-TIME[\( t(n) \)]-HARD[\( n^{O(1)} \)]

Quantifiers are parallel: If array \( A[x] : x = 1, \ldots, r \) in memory, \( \forall x(A(x)) \equiv \text{write}(1) \); proc \( p_i: \text{if } (A[i] = 0) \text{ then write}(0) \)
Parallel Machines

\[ \text{CRAM}[t(n)] = \text{CRCW-PRAM-TIME}[t(n)]-\text{HARD}[n^{O(1)}] \]

Quantifiers are parallel: If array \( A[x] : x = 1, \ldots, r \) in memory, \( \forall x (A(x)) \equiv \text{write}(1) \); proc \( p_i : \text{if} (A[i] = 0) \text{ then write}(0) \)
Parallel Machines

CRAM\( [t(n)] = \text{CRCW-PRAM-TIME}[t(n)]-\text{HARD}[n^{O(1)}] \)

Quantifiers are parallel: If array \( A[x] : x = 1, \ldots, r \) in memory, \( \forall x (A(x)) \equiv \text{write}(1); \) proc \( p_i : \text{if} \ (A[i] = 0) \text{ then write}(0) \)
Inductive Definitions

\[ E^*(x,y) \equiv x = y \lor E(x,y) \lor \exists z (E^*(x,z) \land E^*(z,y)) \]
Inductive Definitions

\[ E^*(x, y) \equiv x = y \lor E(x, y) \lor \exists z(E^*(x, z) \land E^*(z, y)) \]

\[ \varphi_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor \exists z(R(x, z) \land R(z, y)) \]

\[ \mathcal{A} \in \text{REACH} \iff \mathcal{A} \models (\text{LFP} \varphi_{tc})(s, t) \]
**Inductive Definitions**

\[ E^*(x, y) \equiv x = y \lor E(x, y) \lor \exists z (E^*(x, z) \land E^*(z, y)) \]

\[ \varphi_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor \exists z (R(x, z) \land R(z, y)) \]

\[ \mathcal{A} \in \text{REACH} \iff \mathcal{A} \models (\text{LFP}\varphi_{tc})(s, t) \]

Thus, \( \text{REACH} \in \text{IND}[\log n] \).
**Inductive Definitions**

\[ E^*(x, y) \equiv x = y \lor E(x, y) \lor \exists z(E^*(x, z) \land E^*(z, y)) \]

\[ \varphi_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor \exists z(R(x, z) \land R(z, y)) \]

\[ A \in \text{REACH} \iff A \models (\text{LFP} \varphi_{tc})(s, t) \]

Thus, \( \text{REACH} \in \text{IND}[\log n] \).

Next, we’ll show that \( \text{REACH} \in \text{FO}[\log n] \).
\( \varphi_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor (\exists z)(R(x, z) \land R(z, y)) \)

1. Dummy universal quantification for base case:

\[
\varphi_{tc}(R, x, y) \equiv (\forall z. M_1)(\exists z)(R(x, z) \land R(z, y))
\]

\[
M_1 \equiv \neg(x = y \lor E(x, y))
\]
\[ \varphi_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor (\exists z)(R(x, z) \land R(z, y)) \]

1. Dummy universal quantification for base case:

\[ \varphi_{tc}(R, x, y) \equiv (\forall z. M_1)(\exists z)(R(x, z) \land R(z, y)) \]
\[ M_1 \equiv \neg(x = y \lor E(x, y)) \]

2. Using \( \forall \), replace two occurrences of \( R \) with one:

\[ \varphi_{tc}(R, x, y) \equiv (\forall z. M_1)(\exists z)(\forall uv. M_2)R(u, v) \]
\[ M_2 \equiv (u = x \land v = z) \lor (u = z \land v = y) \]
φ_{tc}(R, x, y) \equiv x = y \lor E(x, y) \lor (\exists z)(R(x, z) \land R(z, y))

1. Dummy universal quantification for base case:

φ_{tc}(R, x, y) \equiv (\forall z.M_1)(\exists z)(R(x, z) \land R(z, y))
\quad M_1 \equiv \neg(x = y \lor E(x, y))

2. Using ∀, replace two occurrences of R with one:

φ_{tc}(R, x, y) \equiv (\forall z.M_1)(\exists z)(\forall uv.M_2)R(u, v)
\quad M_2 \equiv (u = x \land v = z) \lor (u = z \land v = y)

3. Requantify x and y.

M_3 \equiv (x = u \land y = v)

φ_{tc}(R, x, y) \equiv (\forall z.M_1)(\exists z)(\forall uv.M_2)(\exists xy.M_3)R(x, y)
Define the quantifier block:

\[ QB_{tc} \equiv (\forall z. M_1)(\exists z)(\forall uv. M_2)(\forall xy. M_3) \]

Operator \( \varphi_{tc} \) is equivalent to \( QB_{tc} \):

\[ \varphi_{tc}(R, x, y) \equiv [QB_{tc}]R(x, y) \]

Thus, for any \( r \), \( \varphi_{tc}(\emptyset) \equiv [QB_{tc}]^r(\text{false}) \)

Thus, for any structure \( A \in \text{STRUC}[\tau_g] \),

\[ A \in \text{REACH} \iff A \models (LFP \varphi_{tc})(s, t) \]

\[ \iff A \models ([QB_{tc}]^{1+\log \|A\|} \text{false})(s, t) \]
\textbf{CRAM}[t(n)] = \text{concurrent parallel random access machine; polynomial hardware, parallel time } O(t(n))

\textbf{IND}[t(n)] = \text{first-order, depth } t(n) \text{ inductive definitions}

\textbf{FO}[t(n)] = t(n) \text{ repetitions of a block of restricted quantifiers:}

\begin{equation*}
QB = [(Q_1x_1.M_1) \cdots (Q_kx_k.M_k)]; \quad M_i \text{ quantifier-free}
\end{equation*}

\begin{equation*}
\phi_n = \underbrace{[QB][QB] \cdots [QB]}_{t(n)} M_0
\end{equation*}
Thm: For all constructible, polynomially bounded $t(n)$,

$$\text{CRAM}[t(n)] = \text{IND}[t(n)] = \text{FO}[t(n)]$$

Thm: For all $t(n)$, even beyond polynomial,

$$\text{CRAM}[t(n)] = \text{FO}[t(n)]$$
Thm: For $v = 1, 2, \ldots$, \quad \text{DSPACE}[n^v] = \text{VAR}[v + 1]$

Number of variables corresponds to amount of hardware.

Since variables range over a universe of size $n$, a constant number of variables can specify a polynomial number of gates:

A bounded number of variables corresponds to polynomially much hardware.
Recursive

Primitive Recursive

EXPTIME

PSPACE

Polynomial–Time Hierarchy

NP ∩ co–NP

P

"truly feasible"

NC

NC^2

log(CFL)  SAC^1

NSPACE[log n]

DSPACE[log n]

Regular

NC^1

ThC^0

AC^0

Descriptive Complexity and Nested Words – 14/46
Recursive

Primitive Recursive

EXPTIME

PSPACE

DSPACE[log n]

FO(DTC)

AC

FO

Logarithmic–Time Hierarchy

NSPACE[log n]

SO

NP

SO

Polynomial–Time Hierarchy

SO[2^{n^{O(1)}}]

SO\[n^{O(1)}\]

FO\[2^{n^{O(1)}}\]

P

"truly feasible"

FO\[n^{O(1)}\]

NC

NC^2

log(CFL)

SAC^1

NSPACE[log n]

DSPACE[log n]

AC^0

Descriptive Complexity and Nested Words – 18/46
Descriptive Complexity and Nested Words – 19/46
We Need an Ordering on the Universe

\[ G = (V, E); \quad V = \{0, 1, \ldots, n - 1\}; \quad 0 < 1 < \cdots < n - 1 \]

An unordered graph makes sense mathematically, but you can’t store such an object in a computer as far as I know.

If you remove the ordering then the first-order descriptive characterizations fail:

**EVEN** requires \( \Omega(n) \) variables without ordering.

Thus, **EVEN** \( \not\in \text{FO}(\text{wo} \leq)[2^{n^{O(1)}}]; \quad (\text{FO}[2^{n^{O(1)}}] = \text{PSPACE}) \)

Fagin (\( \text{SO}\exists = \text{NP} \)) didn’t run into this problem because in \( \text{SO}\exists \) you can guess an ordering (unless we are dealing with \( \text{SO}\exists(\text{monadic}) \)).
Ehrenfeucht-Fraïssé Games

**Delilah:** hide any differences between the two structures

Two-person combinatorial game for characterizing what is expressible in a given quantifier depth.

**Fundamental Thm:** \( D \) has a winning strategy on the \( m \)-move, \( k \)-pebble game on \( A, B \) iff \( A \) and \( B \) agree on all formulas using \( k \) variables and quantifier depth \( m \).

\[
A \sim^k_m B \iff A \equiv^k_m B
\]

**Samson:** show a difference
$\text{dist}(x, y) \leq 2^k \in \text{FO}_k^3 \not\in \text{FO}_{k-1}$

**move 0**

\begin{align*}
&x \\
&\downarrow \\
&\bullet \quad \bullet \quad \ldots \quad \bullet \quad \ldots \quad \bullet \\
&0 \quad 1 \quad \ldots \quad 2^{k-2} \quad \ldots \quad 2^{k-1} \quad \ldots \quad 2^{k-1} + 2^{k-2} \quad \ldots \quad 2^k \quad 2^k + 1 \\
&\bullet \quad \bullet \quad \ldots \quad \bullet \quad \ldots \quad \bullet \\
&\uparrow \\
&y
\end{align*}
\[ \text{dist}(x, y) \leq 2^k \quad \in \text{FO}_k^3 \quad \notin \text{FO}_{k-1} \]

**move 1**

\[ \exists z \left( \text{dist}(x, z) \leq 2^{k-1} \land \text{dist}(z, y) \leq 2^{k-1} \right) \]
\[ \text{dist}(x, y) \leq 2^k \quad \in \text{FO}_k^3 \quad \not\in \text{FO}_{k-1} \]

**move 1**  
\[ \exists z (\text{dist}(x, z) \leq 2^{k-1} \land \text{dist}(z, y) \leq 2^{k-1}) \]
\[
\text{move 2} \quad \exists x (\text{dist}(z, x) \leq 2^{k-2} \land \text{dist}(x, y) \leq 2^{k-2})
\]
dist(x, y) ≤ 2^k \in \text{FO}_k^3 \not\in \text{FO}_{k-1}

move 2 \quad \exists x (\text{dist}(z, x) \leq 2^{k-2} \land \text{dist}(x, y) \leq 2^{k-2})
\[ \text{dist}(x, y) \leq 2^k \quad \in \text{FO}_k^3 \quad \not\in \text{FO}_{k-1} \]

**move k-1: Delilah wins**

\[ \text{dist}(x, y) \leq 2^1 \]

\[ \begin{array}{cccccccccccc}
0 & 1 & \ldots & 2^{k-2} & \ldots & 2^{k-1} & \ldots & \ldots & 2^k & 2^k + 1 \\
\end{array} \]
\[
\text{dist}(x, y) \leq 2^k \quad \in \mathbf{FO}_k^3 \quad \not\in \mathbf{FO}_{k-1}
\]

\textbf{move k: Samson wins} \quad E(x, z) \land E(z, y)

\[
\begin{array}{cccccccccccccccccc}
\cdot & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot \\
0 & 1 & \cdots & 2^{k-2} & \cdots & 2^{k-1} & \cdots & \cdots & 2^k & 2^k + 1 \\
\cdot & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot & \cdots & \cdot \\
\end{array}
\]
These games have been used to prove many beautiful lower bounds without the ordering, and thus separate most descriptive classes without ordering. Among others:

- McCollm and Grädel: DTC, TC, LFP
- Grohe: arity hierarchies for TC and LFP
- Etessami: lower bounds with counting and one-way local orderings
- Libkin: lower bounds with counting via locality arguments
- Grohe and Schwentick: locality of order-independent queries
One Historical Thread

- Fagin: $\text{REACH} \not\in \text{SO} \exists \text{(monadic)}$ and thus since $\text{REACH} \in \text{SO} \exists \text{(monadic)}$, $\text{SO} \exists \text{(monadic)}$ is not closed under complementation.

- de Rougemont: remains true with successor

- Schwentick: remains true with ordering!

- Doesn’t give us a complexity lower bound, cf., Lynch: $\text{NTIME}[n^{k}] \subseteq \text{SO} \exists (+) (\text{arity } k)$
EHRENFEUCHT-FRAÎSSÉ GAMES: LOWER BOUNDS ON DEPTH.

WITH ORDERING, \( \text{DEPTH} \ 2 + \log n \) SUFFICES TO EXPRESS ANY GRAPH PROPERTY FOR GRAPHS ON \( n \) VERTICES. LET \( G = (V, E) \).

\( \#_i(x) \equiv \text{"exists a path of length } i \text{ from 0 to } x\" \in \text{DEPTH}(\log n) \)

\[ \varphi_G \equiv \bigwedge_{\langle i, j \rangle \in E} \exists x, y (\#_i(x) \land \#_j(y) \land E(x, y)) \land \bigwedge_{\langle i, j \rangle \notin E} \exists x, y (\#_i(x) \land \#_j(y) \land \neg E(x, y)) \]

FOR \( S \) AN ARBITRARY SET OF GRAPHS ON \( n \) VERTICES:

\[ \varphi_S \equiv \bigvee_{G \in S} \varphi_G; \quad \text{DEPTH}(\varphi_S) = 2 + \log n \]
Newish Size Lower Bound Game

Adler & I defined these games and proved a pair of optimal succinctness result for temporal logics.

Karchmer and Wigderson used similar communication complexity games to prove lower bound on monotone circuits for REACH.

Idea: unlike Ehrenfeucht-Fraïssé games, we play on a pair of sets of structures, $A, B$. We determine the size of the smallest sentence true of all of $A$ and none of $B$.

More recent lower bounds by Grohe and Schweikardt
Newish Size Lower Bound Game

Adler & I defined these games and proved a pair of optimal succinctness result for temporal logics.

Karchmer and Wigderson used similar communication complexity games to prove lower bound on monotone circuits for REACH.

Idea: unlike Ehrenfeucht-Fraïssé games, we play on a pair of sets of structures, $A, B$. We determine the size of the smallest sentence true of all of $A$ and none of $B$.

More recent lower bounds by Grohe and Schweikardt

Not surprisingly, these games are much harder to play than Ehrenfeucht-Fraïssé games . . .
Size versus Number of Variables on Line Graphs

\[
\text{dist}(x, y) \leq 1 \equiv x = y \lor E(x, y)
\]

\[
\text{dist}(x, y) \leq 2k \equiv \exists z (\text{dist}(x, z) \leq k \land \text{dist}(z, y) \leq k)
\]

\[
\text{dist}(x, y) \leq n \in \text{FO}_{\log n}; \quad \text{SIZE}(n)
\]
Size versus Number of Variables on Line Graphs

\[ \text{dist}(x, y) \leq 1 \equiv x = y \lor E(x, y) \]
\[ \text{dist}(x, y) \leq 2k \equiv \exists z (\text{dist}(x, z) \leq k \land \text{dist}(z, y) \leq k) \]
\[ \text{dist}(x, y) \leq n \in \text{FO}_{\log n}; \quad \text{SIZE}(n) \]

\[ \text{dist}_u(x, y) \leq 1 \equiv x = y \lor E(x, y) \lor E(y, x) \]
\[ \text{dist}_u(x, y) \leq 2k \equiv \exists z \forall w ((w = x \lor w = y) \rightarrow \text{dist}_u(z, w) \leq k) \]
\[ \text{dist}_u(x, y) \leq n \in \text{FO}_{\log n}; \quad \text{SIZE}(O(\log n)) \]
Size versus Number of Variables on Line Graphs

\[ \text{dist}(x, y) \leq 1 \equiv x = y \vee E(x, y) \]
\[ \text{dist}(x, y) \leq 2k \equiv \exists z(\text{dist}(x, z) \leq k \land \text{dist}(z, y) \leq k) \]
\[ \text{dist}(x, y) \leq n \in \text{FO}_{3 \log n}; \quad \text{SIZE}(n) \]

\[ \text{dist}_u(x, y) \leq 1 \equiv x = y \vee E(x, y) \vee E(y, x) \]
\[ \text{dist}_u(x, y) \leq 2k \equiv \exists z \forall w((w = x \lor w = y) \rightarrow \text{dist}_u(z, w) \leq k) \]
\[ \text{dist}_u(x, y) \leq n \in \text{FO}_{4 \log n}; \quad \text{SIZE}(O(\log n)) \]

Grohe-Schweikardt:
- \text{FO}^2 \text{ and } \text{FO}^3 \text{ are polynomially SIZE-related}
- Exponential Size gap between \text{FO}^3 \text{ and } \text{FO}^4
- Gap between \text{FO}^k \text{ and } \text{FO}^{k+1} \text{ is open for } k > 3
**FO$^2$ on Words: [Philipp Weis & I]**

- FO on words reduces to FO$^3$ on words
- FO$^2$ on words was well studied, except alternation hierarchy was open
**FO² on Words: [Philipp Weis & I]**

- FO on words reduces to $\text{FO}^3$ on words
- $\text{FO}^2$ on words was well studied, except alternation hierarchy was open

**Rankers**

```
b a c d b b a c d
```
**FO² on Words: [Philipp Weis & I]**

- **FO** on words reduces to **FO³** on words
- **FO²** on words was well studied, except alternation hierarchy was open

**Rankers**

\[
\begin{array}{cccccccc}
  b & a & c & d & b & b & a & c & d \\
\end{array}
\]

\[
\uparrow
\]

\[
\triangleleft a \\
1 \text{ ranker}
\]
**FO\(^2\) on Words: [Philipp Weis & I]**

- **FO** on words reduces to **FO\(^3\)** on words
- **FO\(^2\)** on words was well studied, except alternation hierarchy was open

**Rankers**

```
  b  a  c  d  b  b  a  c  d  ▶a  ▶b
                              ▲
                      2 ranker
```
**FO\(^2\) on Words: [Philipp Weis & I]**

- **FO** on words reduces to **FO\(^3\)** on words
- **FO\(^2\)** on words was well studied, except alternation hierarchy was open

**Rankers**

\begin{align*}
\begin{array}{cccccccc}
 b & a & c & d & b & b & a & c & d \\
\uparrow & & & & & & & & \\
\end{array}
\end{align*}

\begin{align*}
\triangleright a & \triangleright b & \triangleleft c \\
3 \text{ ranker}
\end{align*}
\( \text{FO}^2 \) on Words: [Philipp Weis & I]

- \( \text{FO} \) on words reduces to \( \text{FO}^3 \) on words
- \( \text{FO}^2 \) on words was well studied, except alternation hierarchy was open

Rankers

\[
\begin{array}{ccccccc}
  b & a & c & d & b & b & a & c & d \\
\end{array}
\]

\[
\begin{array}{cccc}
  \triangleright a & \triangleright b & \triangleleft c & \triangleleft d \\
  4 \text{ ranker}
\end{array}
\]
**FO$^2$ on Words: [Philipp Weis & I]**

- **FO** on words reduces to **FO$^3$** on words
- **FO$^2$** on words was well studied, except alternation hierarchy was open

**Rankers**

\[
\begin{array}{cccccccc}
 b & a & c & d & b & b & a & c & d \\
\uparrow & & & & & & & &
\end{array}
\]

\[\triangleright a \quad \triangleright b \quad \triangleleft c\]

3 ranker

**Thm:** $\text{FO}_m^2 \equiv \text{FO}_m$ which $m$-rankers exist, and, relative order with smaller rankers.
\( \text{FO}^2 \) on Words: [Philipp Weis & I]

- \( \text{FO} \) on words reduces to \( \text{FO}^3 \) on words
- \( \text{FO}^2 \) on words was well studied, except alternation hierarchy was open

**Rankers**

\[
\begin{array}{ccccccc}
  b & a & c & d & b & b & a & c & d \\
\end{array}
\]

\[\uparrow\]

\[\triangleright a \triangleright b \triangleleft c\]

2, 3 ranker

**Thm:** \( \text{FO}_m^2 \equiv \) which \( m \)-rankers exist, and, relative order with smaller rankers.

**Thm:** \( \text{FO}_{k,m}^2 \equiv \) which \( k, m \)-rankers exist, and, relative order with smaller rankers.
FO$^2$ on Words: [Philipp Weis & I]

- FO on words reduces to FO$^3$ on words
- FO$^2$ on words was well studied, except alternation hierarchy was open

**Rankers**

\[ b \ a \ c \ d \ b \ b \ a \ c \ d \]

\[ \uparrow \]

\[ \triangleright a \ \triangleright b \ \triangleleft c \]

**Thm:** FO$_m^2 \equiv$ which $m$-rankers exist, and, relative order with smaller rankers.

**Thm:** FO$_{k,m}^2 \equiv$ which $k, m$-rankers exist, and, relative order with smaller rankers.

**Thm:** FO$^2$ has a strict alternation hierarchy.
Ultimately, want to understand the VAR-SIZE trade-off in many settings.

Currently, Philipp Weis and I would be happy to settle what Grohe and Schweikardt left open, then to move on to words, and other simple graphs, 2 and 3 variables . . .
Ultimately, want to understand the VAR-SIZE trade-off in many settings.

Currently, Philipp Weis and I would be happy to settle what Grohe and Schweikardt left open, then to move on to words, and other simple graphs, 2 and 3 variables . . .

Would one day like to understand the following:

\[ \text{PSPACE} = \text{FO}[2^{n^{O(1)}}] = \text{SO}[n^{O(1)}] \]

\[ \text{NC}^1 \subseteq \text{FO}[\log n/ \log \log n] \]

\[ \text{NC}^1 \subseteq \text{L} \subseteq \text{NL} \subseteq \text{sAC}^1 \subseteq \text{AC}^1 = \text{FO}[\log n] \]
I’ll survey work by Alur, Madhusudan, Kumar, Viswanathan, Arenas, Barceló, Etessami, and Libkin [LICS 2007].
Nested Words

words on some alphabet, $\Sigma$
well-nested edges
Three kinds of nodes: call, return, and internal
Nested Words

- words on some alphabet, $\Sigma$
- well-nested edges
- Three kinds of nodes: call, return, and internal

Applications:
- calls and returns: model checking recursive programs
- parse trees: linguistics
- XML
Nested Words

- words on some alphabet, $\Sigma$
- well-nested edges
- Three kinds of nodes: call, return, and internal

Applications:
- calls and returns: model checking recursive programs
- parse trees: linguistics
- XML

“CFL’s for the price of regular languages”
Nested Word Automata: NWA

An NWA, \((Q, \Sigma, \delta, s, F)\), is just like a finite automaton except, at return nodes, the new state depends on both previous states.

For example, \(q_4 = \delta_r(q_1, q_3, 0)\)
Example of a Regular Nested Language

The balanced parentheses languages of nested words, in which the nesting edges go from each open parenthesis to its mate, are regular.

\[
\begin{array}{c}
[ \\
( a ) \\
( a ) \\
] \\
[ ] \\
a
\end{array}
\]
Regular Nested Languages

- Closed under $\cup$, $\cap$, $\ast$, concatenation, complementation.
- For every $n$-state nondeterministic NWA, there is an equivalent deterministic NWA with at most $2^{n^2}$ states.
- EMPTY-NWA, minimal deterministic NWA, union intersection, language containment for deterministic NWA are all polynomial time.
- Language containment given nondeterministic NWA’s is exptime complete.

Compare this with CFL’s where containment is undecidable.
\((u, i) \oplus v = \text{insert } v \text{ after letter } i \text{ in } u.\)
\[(u, i) \oplus v = \text{insert } v \text{ after letter } i \text{ in } u.\]
\[(u, i) \oplus v = \text{insert } v \text{ after letter } i \text{ in } u.\]
Myhill Nerode Theorem for Nested Words

\[ x \sim_\mathcal{L} y \iff \forall u, i \ ( (u, i) \oplus x \in \mathcal{L} \leftrightarrow (u, i) \oplus y \in \mathcal{L) } \]
Myhill Nerode Theorem for Nested Words

\[ x \sim_{\mathcal{L}} y \text{  iff  } \forall u, i \ ( (u, i) \oplus x \in \mathcal{L} \iff (u, i) \oplus y \in \mathcal{L} ) \]

**Thm:** [Alur, Kumar, Madhusudan, and Viswanathan]

The nested word language, \( \mathcal{L} \), is regular iff \( \sim_{\mathcal{L}} \) has finitely many equivalence classes.
Notation

\[ \mu(i,j): \quad i \text{ is a call node, and } j \text{ is its matching return} \]

\[ r(i) = j \quad c(j) = i \]
Notation

\[ \mu(i, j) : \quad i \text{ is a call node, and } j \text{ is its matching return} \]

\[ r(i) = j \quad c(j) = i \]

\[ C(x) \text{ is the innermost call that } x \text{ is within, e.g., } \]
\[ C(i) = C(j) = a; \quad C(m) = i \]
Notation

\( \mu(i, j) : \quad i \) is a call node, and \( j \) is its matching return

\[ r(i) = j \quad \quad c(j) = i \]

\( C(x) \) is the innermost call that \( x \) is within, e.g.,

\[ C(i) = C(j) = a; \quad C(m) = i \]

\( R(x) \) is the innermost return that \( x \) is within, e.g.,

\[ R(i) = R(j) = z; \quad R(m) = j \]
Monadic Second-Order Logic, (MSO$_\mu$)

$\varphi := Q_a(x) \mid x \in X \mid x \leq y \mid \mu(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x(\varphi) \mid \exists X(\varphi)$
Monadic Second-Order Logic, \((\text{MSO}_\mu)\)

\[\varphi := Q_a(x) \mid x \in X \mid x \leq y \mid \mu(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x(\varphi) \mid \exists X(\varphi)\]

**Thm:** [Alur and Madhusudan]

\[\text{MSO}_\mu = \text{Regular Nested Languages}\]
Monadic Second-Order Logic, (MSO$_\mu$)

$\varphi := Q_a(x) \mid x \in X \mid x \leq y \mid \mu(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x(\varphi) \mid \exists X(\varphi)$

Thm: [Alur and Madhusudan]

$\text{MSO}_\mu = \text{Regular Nested Languages}$

Proof: Similar to proof that MSO = Regular.
Monadic Second-Order Logic, (MSO$_\mu$)

$\varphi := Q_a(x) \mid x \in X \mid x \leq y \mid \mu(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x(\varphi) \mid \exists X(\varphi)$

Thm: [Alur and Madhusudan]

MSO$_\mu = \text{Regular Nested Languages}$

Proof: Similar to proof that MSO = Regular.

MSO $\subseteq$ Regular: new case is $\mu(c, r)$. An NWA can check that the call related to $r$ was $c$. 
Monadic Second-Order Logic, (MSO$_\mu$)

$\varphi ::= Q_a(x) \mid x \in X \mid x \leq y \mid \mu(x, y) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x(\varphi) \mid \exists X(\varphi)$

Thm: [Alur and Madhusudan]

$$\text{MSO}_\mu = \text{Regular Nested Languages}$$

Proof: Similar to proof that MSO = Regular.

MSO $\subseteq$ Regular: new case is $\mu(c, r)$. An NWA can check that the call related to $r$ was $c$.

Regular $\subseteq$ MSO: as usual, in MSO we guess the accepting run of the automaton. New case: we use $\mu$ to check that the return transitions are correct.
Temporal Logic for Nested Words

Next Operators:

\((w, i) \models \bigcirc \varphi \iff (w, i + 1) \models \varphi\)

\((w, i) \models \bigcirc \mu \varphi \iff \exists j \left( \mu(i, j) \land (w, j) \models \varphi \right)\)

Past Operators:

\((w, j) \models \ominus \varphi \iff (w, j - 1) \models \varphi\)

\((w, j) \models \ominus \mu \varphi \iff \exists i \left( \mu(i, j) \land (w, i) \models \varphi \right)\)

Since and Until Operators for several kinds of paths:

\((w, i) \models \alpha U \beta \iff \exists \text{ path } i = i_0 < i_1 < \cdots < i_k\)
\( (w, i_k) \models \beta \land \forall j < k (w, i_j) \models \alpha \)

\((w, i) \models \alpha S \beta \iff \exists \text{ path } i = i_0 > i_1 > \cdots > i_k\)
\( (w, i_k) \models \beta \land \forall j < k (w, i_j) \models \alpha \)
4 Kinds of Paths

The sequence, \( i = i_0 < i_1 < \cdots < i_k = j \) is a linear path if \( i_{p+1} = i_p + 1 \)
4 Kinds of Paths

The sequence, \( i = i_0 < i_1 < \cdots < i_k = j \) is

- a **linear path** if \( i_{p+1} = i_p + 1 \)
- a **call path** if \( i_p = C(i_{p+1}) \)
4 Kinds of Paths

The sequence, \( i = i_0 < i_1 < \cdots < i_k = j \) is

- a **linear path** if \( i_{p+1} = i_p + 1 \)
- a **call path** if \( i_p = C(i_{p+1}) \)
- an **abstract path** if \( i_{p+1} = \begin{cases} r(i_p) & \text{if } i_p \text{ is a call} \\ i_p + 1 & \text{otherwise} \end{cases} \)
4 Kinds of Paths

The sequence, \( i = i_0 < i_1 < \cdots < i_k = j \) is

- a **linear path** if \( i_{p+1} = i_p + 1 \)
- a **call path** if \( i_p = C(i_{p+1}) \)
- an **abstract path** if \( i_{p+1} = \begin{cases} r(i_p) & \text{if } i_p \text{ is a call} \\ i_p + 1 & \text{otherwise} \end{cases} \)
- a **summary path** if \( i_{p+1} = \begin{cases} r(i_p) & \text{if } i_p \text{ is a call and } r(i_p) \leq j \\ i_p + 1 & \text{otherwise} \end{cases} \)
4 Resulting Kinds of Until and Since Operators

- **linear paths:** $U$, $S$
- **call paths:** $U^c$, $S^c$
- **abstract paths:** $U^a$, $S^a$
- **summary paths:** $U^\sigma$, $S^\sigma$

$$(w, i) \models \alpha U \beta \iff \exists \text{ path } i = i_0 < i_1 < \cdots < i_k$$
$$(w, i) \models \alpha S \beta \iff \exists \text{ path } i = i_0 > i_1 > \cdots > i_k$$
Nested Word Temporal Logic: NWTL

\[ \varphi \ := \ a \ | \ \neg \varphi \ | \ \varphi \lor \varphi \ | \ \Box \varphi \ | \ \Box \mu \varphi \ | \ \Diamond \varphi \ | \ \Diamond \mu \varphi \ | \ \varphi \mathbf{U}^\sigma \varphi \ | \ \varphi \mathbf{S}^\sigma \varphi \]
Nested Word Temporal Logic: NWTL

\[ \varphi := a \mid \neg \varphi \mid \varphi \lor \varphi \mid \bigcirc \varphi \mid \bigcirc \mu \varphi \mid \bigcap \varphi \mid \bigcap \mu \varphi \mid \varphi^U \sigma \varphi \mid \varphi^S \sigma \varphi \]

**Thm:** [LICS 07] NWTL is exactly as expressive as FO over finite, and over infinite, nested words.
Nested Word Temporal Logic: NWTL

\[ \varphi ::= a \mid \neg \varphi \mid \varphi \lor \varphi \mid \bigcirc \varphi \mid \bigcirc \mu \varphi \mid \bigcirc \nu \varphi \mid \bigcirc \mu \varphi \mid \bigcirc \nu \varphi \mid \varphi U^\sigma \varphi \mid \varphi S^\sigma \varphi \]

**Thm:** [LICS 07] NWTL is exactly as expressive as FO over finite, and over infinite, nested words.

**Proof:**
NWTL \( \subseteq \) FO: translate each \( \varphi \in \) NWTL to \( \tau(\varphi) \in \) FO\(^3\) s.t.,

\[
(w, i) \models \varphi \iff (w, i/x) \models \tau(\varphi).
\]
Nested Word Temporal Logic: NWTL

\[ \varphi := a \mid \neg \varphi \mid \varphi \lor \varphi \mid \bigcirc \varphi \mid \bigcirc \mu \varphi \mid \bigcirc \varphi \mid \bigcirc \mu \varphi \mid \varphi \U^\sigma \varphi \mid \varphi \S^\sigma \varphi \]

**Thm:** [LICS 07] NWTL is exactly as expressive as FO over finite, and over infinite, nested words.

**Proof:**
NWTL \( \subseteq \) FO: translate each \( \varphi \in \text{NWTL} \) to \( \tau(\varphi) \in \text{FO}^3 \) s.t.,

\[(w, i) \models \varphi \iff (w, i/x) \models \tau(\varphi) .\]

**Cor:** Every formula in FO on nested words is equivalent to a formula in \( \text{FO}^3 \), i.e., using at most three variables, \( x, y, z \).
Nested Word Temporal Logic: NWTL

\[ \varphi ::= a \mid \neg \varphi \mid \varphi \lor \varphi \mid \Diamond \varphi \mid \Diamond \mu \varphi \mid \Box \varphi \mid \Box \mu \varphi \mid \varphi U^\sigma \varphi \mid \varphi S^\sigma \varphi \]

**Thm:** [LICS 07] NWTL is exactly as expressive as FO over finite, and over infinite, nested words.

**Proof:**
NWTL \( \subseteq \) FO: translate each \( \varphi \in \text{NWTL} \) to \( \tau(\varphi) \in \text{FO}^3 \) s.t.,

\[(w, i) \models \varphi \iff (w, i/x) \models \tau(\varphi).\]

FO \( \subseteq \) NWTL: using FO-completeness of a temporal logic on trees [Marx].

**Cor:** Every formula in FO on nested words is equivalent to a formula in \( \text{FO}^3 \), i.e., using at most three variables, \( x, y, z \).
Model Checking

**Thm:** Satisfiability of NWTL is EXPTIME complete.
**Model Checking**

**Thm:** Satisfiability of NWTL is EXPTIME complete.

**Proof:** [Sketch]

**lower bound:** Previously known from Caret Paper [AEM].
Model Checking

**Thm:** Satisfiability of NWTL is EXPTIME complete.

**Proof:** [Sketch]

- **lower bound:** Previously known from Caret Paper [AEM].

- **upper bound:** Transform formula $\varphi$ to a nested-word automaton of size $2^{O(|\varphi|)}$ and test emptiness.

**Cor:** Model checking NWTL specifications wrt “boolean programs” or nested-word automata is EXPTIME complete.
Work To Be Done on Nested Words

- Fill out more of the theory of nested words.
- Use for model checking recursive programs.
- Use for developing better query languages and algorithms for XML.
- Use in Linguistics.