Descriptive Complexity Characterizations

Isik Ulusan

May 07, 2026

Complexity Classes

P: Polynomial Time.
NP: Nondeterministic Polynomial Time.
co-NP: Languages whose complements are in **NP**.
PH: Languages with poly-time ATMs with $O(1)$ alternations.
NL: Nondeterministic Logarithmic Space.
PSPACE: Polynomial Time.

Non-determinism

Nondeterminism allows a TM to make choices, so an input can result in different computation.
A nondeterministic TM (NTM) accepts when at least one computation results in acceptance.

Second Order Logic

In first order logic (FO), quantifiers can only quantify over elements.
In second order logic (SO), quantifiers can also quantify over sets of elements.
This is equivalent to quantifying over predicates. (Why?)

$SO\exists$ will denote SO but the second order quantifiers can only be existential (no $\forall$ over predicates.)
$SO\forall$ is defined similarly.

The First Characterization, Fagin's Theorem

This was the first result that started Descriptive Complexity and how it characterizes standard computation complexity classes.

$$\mathbf{NP} = SO\exists$$

Our aim will be to write descriptive characterizations $\Phi \in L$ and problem P:

$$\mathcal{A} \in P \iff \mathcal{A} \models \Phi$$

[Fa74]

$SO\exists \subseteq$ NP

To show this, it is sufficient to give for each $SO\exists$ sentence ϕ, a poly-time NTM N such that on input $\mathcal{A}$:

$$\mathcal{A} \models \phi \iff \mathcal{A} \in L(N)$$

That is, N accepts iff the $\mathcal{A}$ models ϕ.

$\phi = \exists R_1, \ldots, R_n : \psi$
So N just guesses the $R_1, \ldots R_n$ as bit strings.
Then it tests $\mathcal{A}' \models \phi$ where $\mathcal{A}' = (\mathcal{A}, R_1, \ldots, R_n)$, which can be done in logarithmic space.

Proposition 7.6 [Im99]

$\mathbf{NP} \subseteq SO\exists$

Consider $3SAT \in$ **NP-complete** (under FO reductions).
We encode each ψ in the following model $\mathcal{A}_\psi$:

- $|\mathcal{A}|$ has both $\{0, \ldots, n\}$ for variable indices and $\{0, \ldots, m\}$ for clause indices.
- $P(i, j)$ is true when variable i is positively in clause j.
- $N(i, j)$ is true when variable i is negatively in clause j.

The idea is to say there is a truth assignment T:
$\Phi = \exists T \subseteq [n] : \forall j \subseteq [m] : (\exists i : P(i, j) \wedge T(i)) \vee (\exists i : N(i, j) \wedge \neg T(i))$.

Theorem 7.8 and Proposition 7.17 [Im99]

co-NP $\subseteq SO\forall$

We now know **NP** $= SO\exists$ and take $L \in$ **coNP** where $\overline{L} \in$ **NP**.
So there is a $\Phi = \exists R_1, \ldots, R_n : \varphi \in SO\exists$ such that:

$$\mathcal{A} \models \Phi \iff \mathcal{A} \in \overline{L}$$

So $\mathcal{A} \models \neg\Phi \iff \mathcal{A} \in L$.
Note that $\neg\Phi = \neg\exists R_1, \ldots, R_n : \varphi = \forall R_1, \ldots, R_n : \neg\phi \in SO\forall$.
So $L \in SO\forall$.

$SO\forall \subseteq$ co-NP

Let $L \in SO\forall$ by $\Phi = \forall R_1, \ldots R_n : \varphi$
Then $\overline{L}$ is given by $\neg\Phi = \exists R_1, \ldots, R_n : \neg\varphi \in SO\exists$.
So $\overline{L} \in$ **NP**.

Alternation

Non-determinism can be taught of as existentially choosing paths.
What about universally choosing?
Or a combination of the both?

The Polynomial Hierarchy

PH is the generalization of **NP** and **co-NP**.
Each level has constant alternations.
So every level is characterized by a poly-time TM with c number of alternations.
Each level has the complete language:
$\Sigma_i - 3SAT = \{\phi | \exists x_1 \forall x_2 \cdots Q x_i : \phi = \top\}$

[St75]

$\mathbf{PH} = SO$

$SO \subseteq \mathbf{PH}$: follows from just using alternation to alternate over the second order quantifiers. Then check $\mathcal{A}' \models \Phi$ where $\mathcal{A}' = (R_1, \ldots, R_n, \mathcal{A})$ using the poly-time ATM.

$\mathbf{PH} \subseteq SO$: $\Sigma_k - 3SAT$:
The model $\mathcal{A}$ still has $P(i,j)$ and $N(i,j)$ and the same universe as before.
Now add for any $\ell \in [k]$, $B_\ell(i)$ meaning variable i is is quantified by T_ℓ.
Now $\Phi = \exists T_1 \forall T_2 \cdots Q T_k : \forall j :$
$\bigvee_{\ell=1}^{k} (\exists i : P(i,j) \wedge B_\ell(i) \wedge T(i)_\ell) \vee (\exists i : N(i,j) \wedge B_\ell(i) \wedge \neg T(i)_\ell)$

Section 3 Remark [St75]

Transitive Closure

The transitive closure $TC(\phi)$ is the smallest predicate for a binary predicate ϕ such that $\phi \subseteq TC(\phi)$ (when thought of as sets) and it is transitive.

$$\forall x, y, z : (\phi(x, y) \wedge \phi(y, z)) \rightarrow (TC(\phi)(x, z))$$

So this new $TC(\phi)$ can essentially be used as a predicate. Example: The structure is a graph with an edge relation E, then $TC(E)$ is the path relation.

If a $TC(\phi)$ never appears within a negation, then this is called *pos TC*.

NL

It is the class of problems solvable by an NTM that uses logarithmic space. It has the complete problem *PATH* for s-t

connectivity on directed graphs.

$\mathbf{NL} = FO + pos\ TC$

$FO + pos\ TC \subseteq \mathbf{NL}$: For quantifiers we can just reuse the space to evaluate the correct values. For $TC(\phi)(a, a')$ we can guess a middle node and compute recursively.

$\mathbf{NL} \subseteq FO + pos\ TC$: Consider $PATH \in$ **NL-complete** (under FO reductions).
The structure is a graph with the edge predicate with specific s and t nodes.
So the formulas for this is:

$$\Phi = TC(E)(s, t) \equiv PATH$$

Theorem 3.16 and 9.10 [Im99]

$\mathbf{NL} = FO + TC$

This relies on the Immerman-Szelepcsényi Theorem:

$$\mathbf{NL} = \mathbf{co\text{-}NL}$$

Every $\psi \in FO + pos\ TC$ can be written as $TC(\alpha)(\overline{0}, \overline{\mathsf{max}})$, with α quantifier free. Done by induction on the formula and needs SUC.
So anything in **co-NL** can be written as $\neg TC(\alpha)(\overline{0}, \overline{max})$.

So showing that there is no path .i.e
$\overline{PATH} \in$ **co-NL-complete** is expressible in $FO + pos\ TC$ suffices.
This is immediate from the Immerman-Szelepcsényi Theorem.

Theorem 1 and Corollary 3 [Im88] and Lemma 9.12 and Theorem 9.20 [Im99]

The Normal Form Proof Sketch

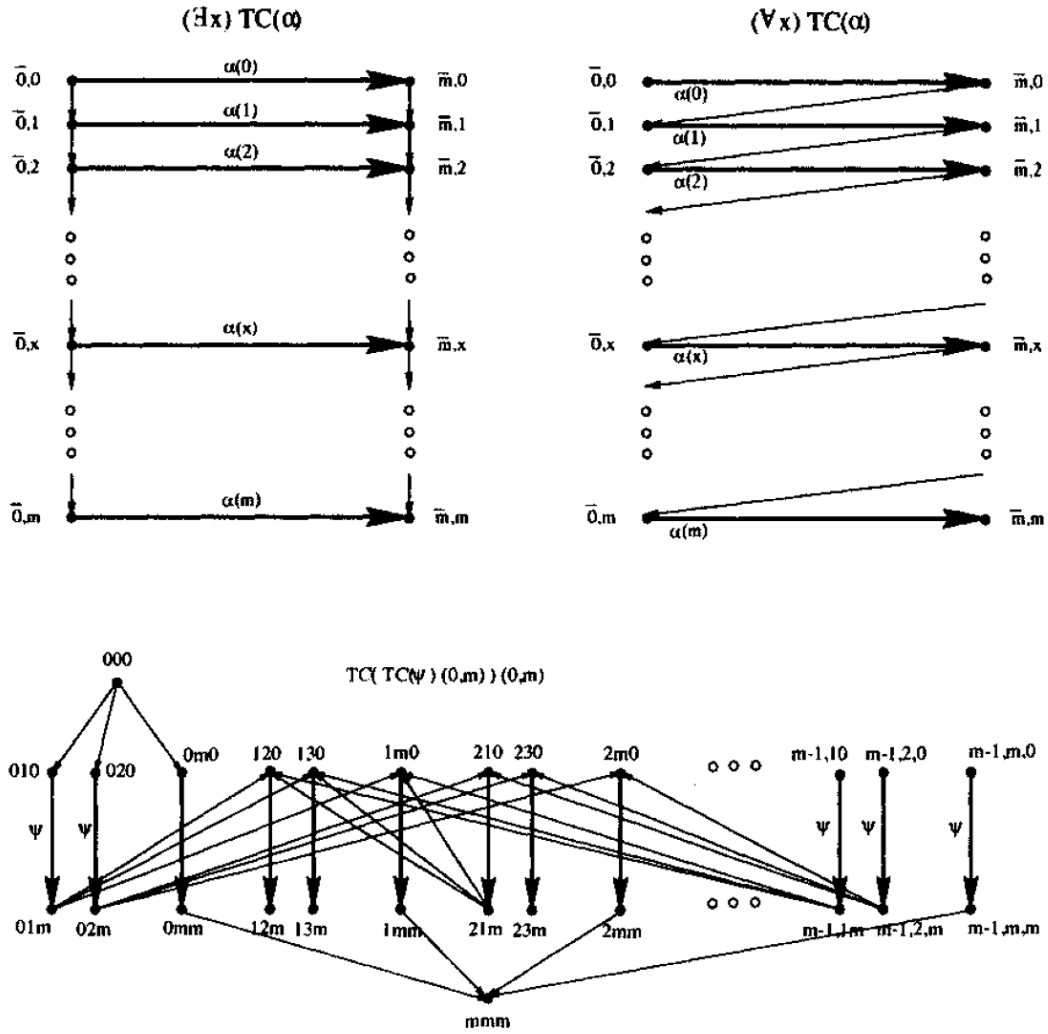


Figure 9.13: Cases 3, 4, and 5 of Lemma 9.12

Lemma 9.12 [Im99]

PSPACE

It is the class of problems solvable by a TM that uses polynomial space.

$TQBF = \{\phi | \phi \in FO$ is bounded and $\models \phi\}$ (it also doesn't have any predicates or functions).
$TQBF \in$ **PSPACE-complete** (under FO reductions).

Proposition 10.2 [Im99]

$SO + TC \subseteq$ **PSPACE**

We have a similar normal form as $TC(\alpha)(\overline{\textbf{false}}, \overline{\textbf{true}})$ where α is quantifier-free, the proof is essentially same as the previous.

From this we a **PSPACE** TM can guess a middle and do divide and conquer, reusing space.
This also relies on **PSPACE** $=$ **NPSPACE** which also uses the same divide and conquer idea.

Exercise 10.23 [Im99] and Theorem 2 [Sa70]

PSPACE $\subseteq SO + TC$

PSPACE $\subseteq SO + TC$: Consider
$TQBF \in$ **PSPACE-complete**.
We want to use TC to extend the assignment through the quantifiers.
The structure is: $\mathcal{A}$

- $|\mathcal{A}| = [n]$ which will be used as indices.
- $Var(i)$, $Clause(i)$, $P(i,j)$, $N(i,j)$, $Exists(i)$, $All(i)$, $\leq (i,j)$, $\mathrm{SUC}(i,j)$.

Now consider $\Phi = \exists T_n \exists T_0 : (Init(T_n) \wedge Base(T_0) \wedge TC[Step((T,k),(T',k'))]((T_n,n),(T_0,0)))$

PSPACE $\subseteq SO + TC$ (cont.)

$Init(T) := \forall b : (T(b) \leftrightarrow Sat(b))$, meaning that T is a satisfying assignment (on the matrix) when $Init(T)$ is true.
$Base(T) := T(\bar{0}, \ldots, \bar{0})$, checking for the base case of the transitive closure.
$Step((T, k), (T', k')) := (k' = k - 1) \wedge Prop(T, T', k)$, which is used to step the *Prop* downwards deeper into the quantifiers.
$Prop(T, T', k)$ essentially means that the assignment T at quantifier depth $k + 1$ is propagated to T' at quantifier depth k (while obeying the quantifier Q_{k+1}). Each T_i (along the

path) should contain the already picked assignments: T_n is the full assignment and T_0 is the empty assignment.
So TC is used to fully backwards propagate T_n to T_0 to verify that T_n yields T_0 through the entire formula.

Transitive Closure vs Least Fixed Points

The transitive closure of a predicate is also a least fixed point.
So $FO + TC \subseteq FO + LFP$.

$LFP(\phi)$ can include elements based on any query (predicate transformer).
However $TC(\phi)$ can only include elements based on the transitivity of a relation.

$\mathbf{P} = FO + LFP$

$FO + LFP \subseteq \mathbf{P}$: To evaluate the $LFP\psi$ for $ar(\psi) = k$ we only need to evaluate it as $(\psi^{\mathcal{A}})^{n^k}(\emptyset)$ where $\|\mathcal{A}\| = n$. So this takes polynomial time. The rest was already in **LOGSPACE** so its in **P** overall.

$\mathbf{P} \subseteq FO + LFP$: $AGAP \leq_{FO} MCVP$ so $MCVP \in$ **P-complete**, then we use $MCVP$:
Let the structure $\mathcal{A}$ be:

- $|\mathcal{A}| = V$, the set of gates.
- $E(i,j)$, $AND(i)$, $OR(i)$, $I_0(i)$, $I_1(i)$, and out.

Define the truth of a gate as:
$T(v) = I_1(v) \vee (OR(v) \wedge \exists u :$
$(E(u,v) \wedge T(u))) \vee (AND(v) \wedge \forall u : (E(u,v) \rightarrow T(u)))$

So $\Phi = LFP(T)(out)$ gives the value of out.

Theorem 1.6 and 4.5[Im87a] and Theorem 4.10 [Im99]

Ordering and $\mathbf{P} = FO + LFP$

This result only works with ordered structures.
So we don't know if $\mathbf{P} = FO + LFP$ in general.

Showing that $MCVP \in$ **P-complete** under FO reductions needs ordering.
(This is also the case when trying to express every language in $\mathbf{P}$ to $FO + LFP$).

Section 3 [Im86] and Section 12.7 [Im99]

References

[Fa74] Fagin - Generalized First-Order Spectra and Polynomial-Time Recognizable Sets

[Im86] Relational Queries Computable in Polynomial Time

[Im87a] Immerman - Languages that Capture Complexity Classes

[Im87b] Immerman - Expressibility as a Complexity Measure: Results and Directions

[Im88] Immerman - Nondeterministic Space is Closed Under Complementation

[Im89] Immerman - Descriptive and Computational Complexity

[Im99] Immerman - Descriptive Complexity

[Sa70] Savitch - Relations Between Nondeterministic and Deterministic Tape Complexities

[St75] Stockmeyer - The Polynomial-Time Hierarchy