Logic and theory of computation

theory of computation part, 6th lecture

(Un)directed Hamiltonian path/cycle

Hamiltonian path/cycle

Given an (un)directed graph $G = (V, E)$ ($|V| = n$). A listing $P = v_{i_1}, \ldots, v_{i_n}$ of the vertices in G is called a **Hamiltonian path** if $\{v_{i_1}, \ldots, v_{i_n}\} = V$ and for all $1 \leq k \leq n-1$ $\{v_{i_k}, v_{i_{k+1}}\} \in E$ ($(v_{i_k}, v_{i_{k+1}}) \in E$ in the undirected case). If $\{v_{i_n}, v_{i_1}\} \in E$ ($(v_{i_n}, v_{i_1}) \in E$ resp.) holds, too than P is called a **Hamiltonian cycle**.

Notation: H-path, H-cycle for Hamiltonian path/cycle.

HP=$\{\langle G, s, t\rangle \mid$ there's an H-path in the directed graph G form s to $t\}$.

UHP=$\{\langle G, s, t\rangle \mid$ there's an H-path in the undirected graph G form s to $t\}$.

UHC=$\{\langle G\rangle \mid G$ is undirected and has an H-cycle$\}$.

NP-completeness of $s-t$-H-path

Theorem

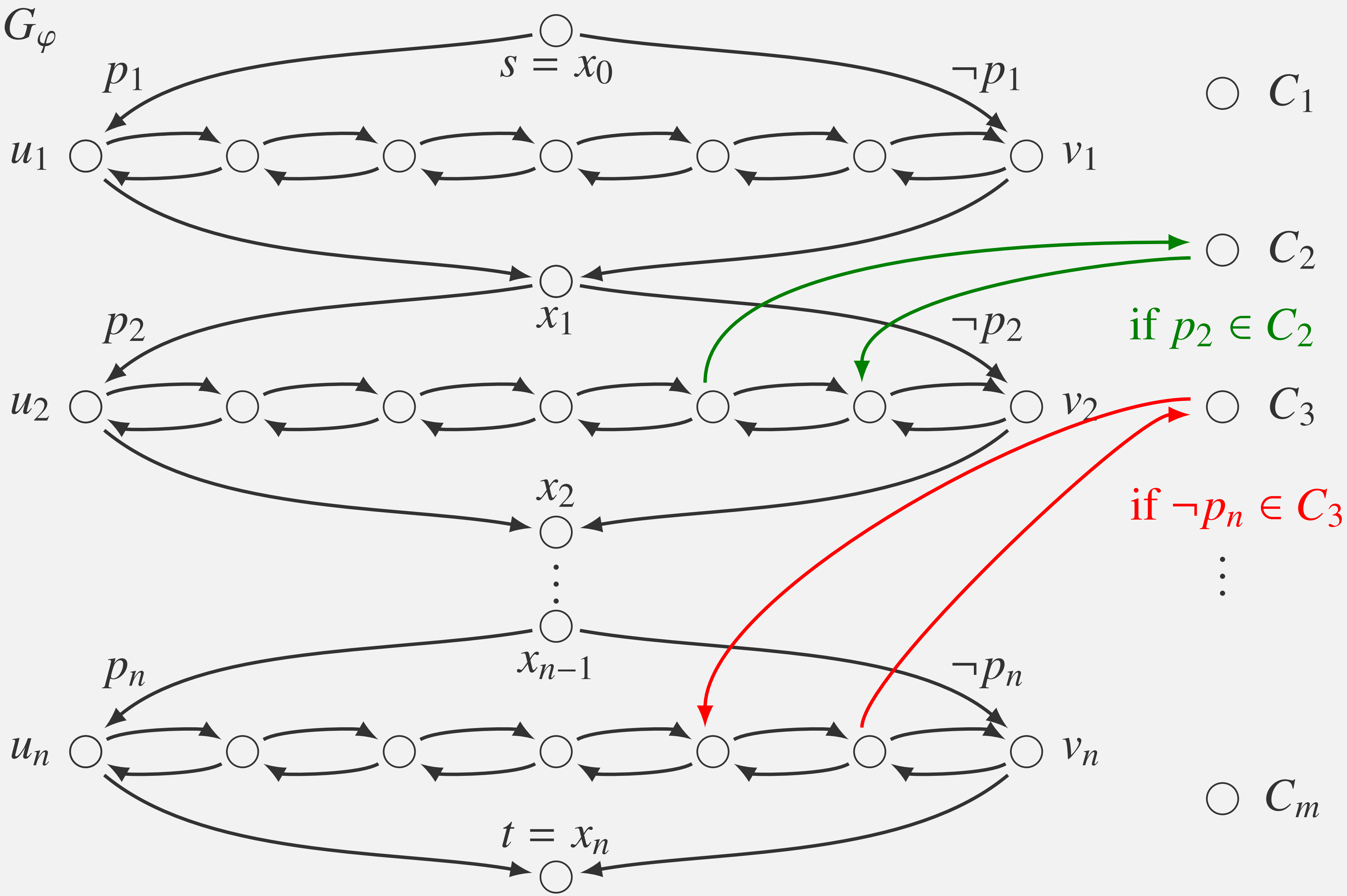
HP is NP-complete

Proof: It's in NP, since a list P of n vertices can be constructed in polynomial time. Being P a permutation of the vertices and an H-path are verifiable in polynomial time, too.

SAT$\leq_p$ HP. For an arbitrary CNF φ it is enough to construct (G_φ, s, t) with the property of φ satisfiable $\Leftrightarrow$ there is an H-path in G_φ from s to t.

Let $p_1, \ldots p_n$ be the atoms appearing in φ and $C_1, \ldots C_m$ be the clauses of φ.

NP-completeness of $s - t$-H-path



NP-completeness of $s - t$-H-path

the construction of G_φ

- $\forall 1 \leq i \leq n : (x_{i-1}, u_i), (x_{i-1}, v_i), (u_i, x_i), (v_i, x_i) :\in E(G_\varphi)$
- $s := x_0, t := x_n$
- $\forall 1 \leq i \leq n$ there is path in both direction with $2m$ inner vertices $w_{i,1}, \ldots, w_{i,2m}$.
- All $w_{i,k}$'s are connected by at most one C_j.
- If $X_i \in C_j$, then $(w_{i,k}, C_j)$ and $(C_j, w_{i,k+1}) :\in E(G_\varphi)$. (positive bound)
- If $\neg X_i \in C_j$, then $(w_{i,k+1}, C_j)$ and $(C_j, w_{i,k}) :\in E(G_\varphi)$. (negative bound)

Positive traversal of $u_i v_i$: $u_i \rightsquigarrow v_i$.
Negative traversal of $u_i v_i$: $u_i \leftsquigarrow v_i$.

NP-completeness of $s-t$-H-path

- An $s-t$ H-path contains (x_{i-1}, u_i) or (x_{i-1}, v_i) but not both $(\forall 1 \le i \le n)$. It must be continued by a positive traversal of $u_i v_i$ in the first case. On the other hand it must be continued by a negative traversal.
- An $s-t$ H-path bounds each C_j exactly one. For a positive traversal of $u_i v_i$, only a positive bound is possible while for a negative one the only option is a negative bound.
- For an H-path, Positive/negative traversals of $u_i v_i$ determine an interpretation, the bound to C_j $(\forall 1 \le j \le m)$ shows a literal in C_j evaluated for true.
- If φ is satisfiable, choose an interpretation and a a true literal for each clause. Choose the positive traversal of $u_i v_i$ if p_i is true, otherwise choose the negative one. Bounding the C_j's to the path at the chosen literals we get an H-path from s to t.

G_φ can be constructed in polynomial time so SAT$\le_p$ HP, proving that HP is NP-hard. But it is in NP, so it is NP-complete as well.

NP-completeness of undirected $s-t$-H-path

Remark: UHP and UHC are in NP by the same reason as HP.

Theorem

UHP NP-complete

Proof: HP$\leq_p$ UHP. For a given G, s, t, where G is directed we need G', s', t', where G' is undirected and G has an H-path from s to $t \Leftrightarrow G'$ has an H-path from s' to t'.
Let 3 nodes correspond to each vertex v of G in G': v_{in}, v_{middle} and v_{out}. Edges of G' contains $\{v_{\text{in}}, v_{\text{middle}}\}$ and $\{v_{\text{middle}}, v_{\text{out}}\}$ for each $v \in V(G)$. Forthermore for each edge $E = (u, v)$ in G add the edge $\{u_{\text{out}}, v_{\text{in}}\}$ to $E(G')$. $s' := s_{\text{in}}, t' := t_{\text{out}}$.
Easy to see that P is an H-path in $G \Leftrightarrow P'$ is an H-path in G', where P' can be constructed for P by replacing v by v_{in}, v_{middle} and v_{out}.

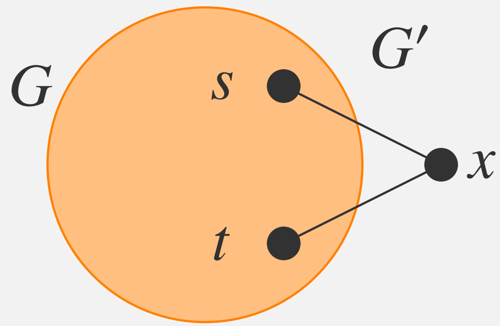
NP-completeness of undirected Hamiltonian path/cycle

On the other hand if P is an H-path in G' than $v_{\text{in}}, v_{\text{middle}}, v_{\text{out}}$ should follow each other in P (otherwise v_{middle} is left out). Replacing these triples in P by a single node v we get an H-path for G.
Conditions for the begining and the ending of paths hold, too.

Theorem

UHC is NP-complete

Proof: UHP$\leq_p$ UHC. Given G, s, t. G': Add a new vertex x to G and add two new edges $\{s, x\}$ and $\{t, x\}$ to G.
Easy to see that G has an H-path $\Leftrightarrow$ G' has an H-cycle.



The travelling salesman problem (TSP)

Function problem: Given an undirected graph G with nonnegative weights on its edges. Determine the H-cycle with the lowest possible total weight (if there's any at all).

Decision problem:

TSP=$\{\langle G, K\rangle \,|\, G$ has an H-cycle with total weights $\leq K\}$.

Theorem

TSP is NP complete.

Proof: TSP$\in$ NP, due to similar reasons as HP (total weight condition is also verifiable in polynomial time).

UHC$\leq_p$ TSP. $G' := G$, all weights are equal to 1 and let $K := |V|$. easy to see, that G has an H-cycle $\Leftrightarrow$ G' has an H-cycle of total weight $\leq K$.

The Structure of the NP class

NP-intermediate language

L is NP-intermediate, if $L \in \mathrm{NP}$, $L \notin \mathrm{P}$ and L is non-NP-complete.

Ladner's Theorem

If $\mathrm{P} \neq \mathrm{NP}$, then there are NP-intermediate languages.

(without proof)

NP-intermediate candidates (we are not sure of course):

- GraphIsomorphism=$\{\langle G_1, G_2\rangle \mid G_1$ and G_2 are undirected isomorphic graphs$\}$.
- PrimeFactors: produce the prime factors of a natural number [function problem],

co$\mathfrak{C}$

co$\mathfrak{C}$ complexity class

If $\mathfrak{C}$ is a complexity class, then let co$\mathfrak{C} = \{L \mid \bar{L} \in \mathfrak{C}\}$.

$\mathfrak{C}$ **is closed for polynomial time reduction** if whenever $L_2 \in \mathfrak{C}$ and $L_1 \leq_p L_2$ holds $L_1 \in \mathfrak{C}$ holds, too.

W know so far: P and NP are closed for polynomial time reduction.

Theorem

If $\mathfrak{C}$ is closed for polynomial time reduction then so does co$\mathfrak{C}$ as well

Proof: Let $L_2 \in$ co$\mathfrak{C}$ and L_1 be arbitrary languages satisfying $L_1 \leq_p L_2$. From the later one it follows $\overline{L}_1 \leq_p \overline{L}_2$ (the same reduction is good). Since $\overline{L}_2 \in \mathfrak{C}$, we get $\overline{L}_1 \in \mathfrak{C}$. I.e., $L_1 \in$ co$\mathfrak{C}$.

co$\mathfrak{C}$

Is it true that P=coP? Yes. (Changing q_a and q_r in a TM deciding L we get a TM deciding $\overline{L}$ polynomial time.)

Is it true, that NP=coNP? The above construction does not work, it may not decide $\overline{L}$.

Corollary

coNP is closed for polynomial time reduction.

Theorem

L is $\mathfrak{C}$-complete $\iff$ $\overline{L}$ is co$\mathfrak{C}$-complete.

Proof:

- If $L \in \mathfrak{C}$, then $\overline{L} \in$ co$\mathfrak{C}$.
- Let $L' \in \mathfrak{C}$, satisfying $L' \leq_p L$. Then $\overline{L'} \leq_p \overline{L}$. $L' \mapsto \overline{L'}$ is a bijection betwwen $\mathfrak{C}$ and co$\mathfrak{C}$. So all co$\mathfrak{C}$ languages are polynomial time reducable to $\overline{L}$

So $\overline{L} \in$ co$\mathfrak{C}$ and it is co$\mathfrak{C}$-hard, so, it is co$\mathfrak{C}$-complete.

Examples for coNP complete languages

UNSAT := $\{\langle\varphi\rangle \mid \varphi$ is an unsatisfiable propositional formula$\}$.

VALID := $\{\langle\varphi\rangle \mid$ a φ is a valid propositional formula $\}$.

Theorem

UNSAT and VALID are coNP-complete.

Proof: GENSAT = $\{\langle\varphi\rangle \mid \varphi$ is a satisfiable propositional formula$\}$ is NP-complete as well.
$\overline{\text{GENSAT}}$ = UNSAT, by the previous Theorem UNSAT is coNP-complete.
UNSAT $\leq_p$ VALID, since $\varphi \mapsto \neg\varphi$ is a polynomial time reduction.

Informally: coNP contains the problems for which there is a polynomial time *refutation* for the "no" instances.

Remarks:
Conjecture: NP $\neq$ coNP.
Another conjecture: P $\neq$ NP $\cap$ coNP.
It is known, that NP $\cap$ coNP-complete $\neq \emptyset$ implies NP = coNP.

Space complexity

Problems with measuring space complexity: The size of the input is always a lower bound for the number of used cells. One solution is the following:

Off-line Turing machine (OTM)

An **off-line Turing machine** (OTM) is a multitape TM with the following properties. The first tape is a read-only input tape. For function problems the last tape is a write-only output tape. Further tapes has no restrictions, they are called storage tapes.

Space complexity of OTM's

For a given input, the **storage space** used up by the OTM is the number of cells the computation(s) use up on the storage tapes. An OTM is $f(n)$ **space-bounded** (or has **space complexity** $f(n)$) if it uses at most $f(|u|)$ storage space for any input $u \in \Sigma^*$.

Deterministic and nondeterministic space complexity classes

With OTM's we can measure **sublinear** space complexities.

- $\mathrm{SPACE}(f(n)) := \{L \mid L$ is decidable by a $O(f(n))$ space-bounded deterministic off-line TM$\}$
- $\mathrm{NSPACE}(f(n)) := \{L \mid L$ is decidable by a $O(f(n))$ space-bounded nondeterministic off-line TM$\}$
- $\mathrm{PSPACE} := \bigcup_{k\geq 1} \mathrm{SPACE}(n^k)$.
- $\mathrm{NPSPACE} := \bigcup_{k\geq 1} \mathrm{NSPACE}(n^k)$.
- $\mathrm{L} := \mathrm{SPACE}(\log n)$.
- $\mathrm{NL} := \mathrm{NSPACE}(\log n)$.

Deterministic space complexity of the REACHABILITY problem

For a (directed) graph G and vertices its x and y, y is called reachable from x, if there is a path from x to y.
REACHABILITY$=\{\langle G, s, t\rangle \mid t$ is reachable in G from $s\}$.
REACHABILITY $\in$ P (in fact $O(n^2)$)

Theorem

REACHABILITY $\in$ SPACE$(\log^2 n)$.

Proof:

- For vertices x, y of G let PATH(x, y, i) be TRUE if there is a path of length at most 2^i from x to y.
- t is reachable in G from $s \iff$ PATH$(s, t, \lceil \log n \rceil)$=TRUE.
- PATH(x, y, i)=TRUE $\iff \exists z($ PATH$(x, z, i-1)$=TRUE $\wedge$ PATH$(z, y, i-1)$=TRUE $)$.
- Based on this observation a recursive algorithm will be given. The algorithm stores only 3-tuples (x, y, i). If (x, y, i) is stored it means PATH(x, y, i) was called and this call is still in progress.

REACHABILITY: PATH(x, y, i) algorithm

- Let an order of the vertices be given.
- If $i = 0$, then $2^0 = 1$, PATH(x, y, i) is TRUE if and only if (x, y) is an edge of the graph.
- We start by writing $(s, t, \lceil \log n \rceil)$ on the storage tape. For the 3-tuples (x, y, i) on the storage tape, i is strictly decreasing (by 1) from $\lceil \log n \rceil$, so the number of 3-tuples is at most $\lceil \log n \rceil + 1$.
- Suppose that we just called PATH(x, y, i), so (x, y, i) is the last 3-tuple on the storage tape. Then for the first $z \neq x, y$ call PATH$(x, z, i - 1)$ and write $(x, z, i - 1)$ on the tape.
- If PATH$(x, z, i - 1)$ =FALSE then delete $(x, z, i - 1)$, try next z.
- If PATH$(x, z, i - 1)$ =TRUE then delete $(x, z, i - 1)$, call PATH$(z, y, i - 1)$ and write $(z, y, i - 1)$ on the tape (y is known from the previous 3-tuple).
- If PATH$(z, y, i - 1)$ =FALSE then delete $(z, y, i - 1)$, try next z.
- If PATH$(z, y, i - 1)$ =TRUE then return TRUE.
- If there are no more z's return FALSE.

Configuration graph

PATH($s, t, \lceil \log n \rceil$) stores $O(\log n)$ 3-tuples and each 3-tuple is of length $O(\log n)$ (we can store a natural number of at most n on $O(\log n)$ bits).
So $\textsc{Reachability} \in \mathrm{SPACE}(\log^2 n)$, finishing the proof.

Configuration graph

For a TM M G_M denotes its **configuration graph**. The vertices are configurations of M and $(C, C') \in E(G_M) \Leftrightarrow C \vdash_M C'$.

Reachability method: a method for proving connections between space complexity classes by using either
$\textsc{Reachability} \in \mathrm{P}$ or
$\textsc{Reachability} \in \mathrm{SPACE}(\log^2 n)$
for the configuration graph or one of its subgraphs.

Savitch's Theorem

Savitch's Theorem

If $f(n) \geq \log n$, then $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$.

Proof: Let M be $f(n)$ space-bounded NTM and w be an input of M of length n.
A configuration of M can be stored at $O(f(n) + \log n)$ cells (actual state, contents of storage tapes, positions of the heads, the position of the head on the 1st tape can be n-wise, that's why we always need $\geq \log n$ storage). If $f(n) \geq \log n$, then this is $O(f(n))$.
We can suppose, that M has only one accepting configuration. (Delete the content of the storage tapes before going to q_a.)
The size of the configuration graph of configurations with this limit is at most $2^{df(n)}$ for some $d > 0$ constant. So there is a$O(\log^2(2^{df(n)})) = O(f^2(n))$ space-bounded deterministic TM deciding reachability from the starting configuration to the accepting one by the previous theorem.

Savitch Theorem

Corollaries

Corollary 1

PSPACE = NPSPACE

Proof: square of a polynomial is a polynomial.

Corollary 2

NL$\subseteq$ P

(without proof) (Applies Reachability method.)

Hierarchy Theorem

Immerman-Szelepcsényi Theorem

NL = coNL

(without proof)

EXPTIME:=$\bigcup_{k\in\mathbb{N}}$TIME(2^{n^k}).

Theorem

NL $\subset$ PSPACE és P $\subset$ EXPTIME.

(without proof)

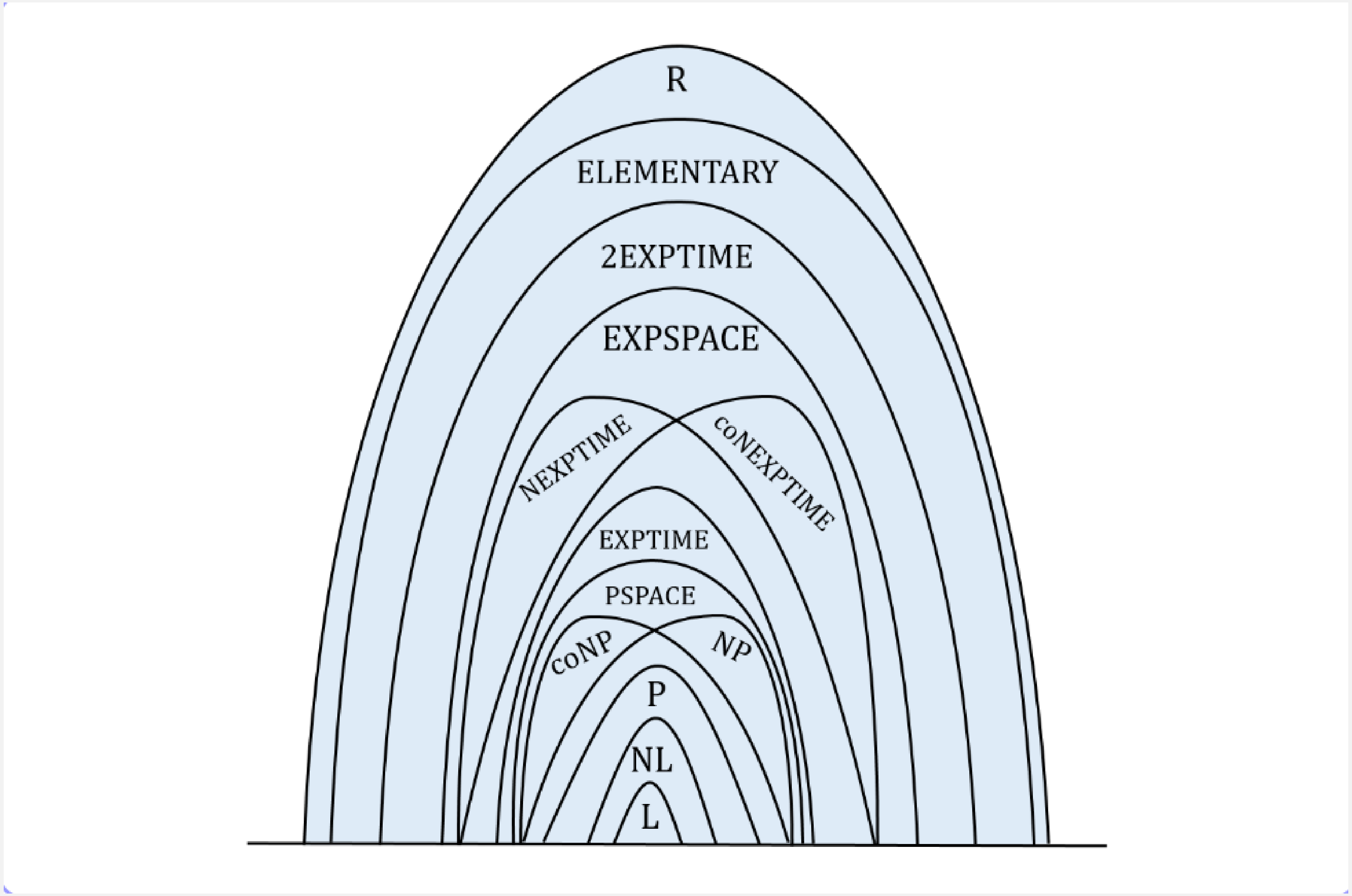
Theorem

L $\subseteq$ NL=coNL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXPTIME

(without proof)

Conjecture: All inclusions are proper.

Structure of R



Structure of R supposing P≠NP