NL and coNL

Xin Li

lixints@cs.jhu.edu
Topics

• Alternative Definition of NL

• NL vs. coNL
Recall: the class \( NL \)

- \( NL = \text{NSPACE}(\log n) \), the class of languages decidable by log space NDTM.

- The definition is by an NDTM.

- Similar to NP, there is an alternative definition based on certificates (the non-deterministic choices of the NDTM).

- Tricky issue: certificate can be polynomially long, cannot be stored by a logspace verifier.
Certificate Based Definition of NL

• Fix: the certificate is provided to the log space verifier on a separate “read once” tape.

• The machine can only read this tape from left to right once.

• The verifier can never read the same bit of the certificate twice.
Certificate Based Definition of NL

• Definition: a language $L$ is in NL if $\exists$ a DTM $M$ (the verifier) with an additional read-once input tape, and a polynomial $p: \mathbb{N} \rightarrow \mathbb{N}$ s.t. $\forall x \in \{0, 1\}^*$,

$$x \in L \iff \exists u \in \{0, 1\}^{p(|x|)} \text{ s.t. } M(x, u) = 1 \text{ (accept)}$$

• $M$ uses at most $O(\log |x|)$ space on its work tapes for every $x$.

• Can show the equivalence to the two definitions.
Equivalence of two Definitions.

• \( NL_1 = \text{NSPACE}(\log n) \), \( NL_2 = \) certificate based definition.

• \( \forall L \in NL_1 \), write the non-deterministic choices of the NDTM on a tape, then the verifier can read them from left to right and follow these choices.

• \( \forall L \in NL_2 \), since the certificate has poly\((n)\) length, the NDTM can guess the certificate bit by bit and use the verifier to check it.

• Note that the NDTM only has log space so it cannot remember previous bits.
The Class coNL

• Similar to coNP, we can define the class coNL.

• $\text{coNL} = \{L \mid \text{the complement of } L \text{ is in } \text{NL} \}.$

• Example: $\neg \text{PATH}=\{<G, s, t>: G \text{ is a directed graph and there is no path in } G \text{ from } s \text{ to } t.\}$

• Since PATH is NL-complete, $\neg \text{PATH}$ is coNL complete ($\forall L \in \text{coNL}, L \leq \text{l} \neg \text{PATH}$).
**NL vs coNL**

- First we can also see $L \subseteq NL \cap coNL$.

- Question: $NL = coNL$? (we believe that $NP \neq coNP$).

- Example: $\neg PATH=\{<G, s, t>: G$ is a directed graph and there is no path in $G$ from $s$ to $t$.\}$

- Seems hard to find any read once certificate for the non-existence of a path from $s$ to $t$. 
**NL vs coNL**

- **Surprise!**

- Theorem (Immerman-Szelepcsényi’ 87): $\neg \text{PATH} \in \text{NL}$.

- **Implication:** $\text{coNL} \subseteq \text{NL}$ and thus $\text{NL} = \text{coNL}$
NL vs coNL

• Theorem (Immerman-Szelepcsenyi’ 87): \( \neg \text{PATH} \in \text{NL} \).

• Proof: use the certificate based definition of NL.

• Show a log space algorithm A s.t. \( \forall \) n-vertex graph G and vertices s, t, 

\[
t \text{is not reachable from } s \text{ iff } \exists \text{ a read once certificate } u \text{ s.t. } A(<G, s, t>, u)=1.
\]
Read Once Certificate for $\neg$ PATH

- Identify $G$’s vertices with the set $\{1, 2, \ldots, n\}$.

- Goal: a read once certificate for the fact that $t$ is not reachable from $s$.

- Seems hard at first, so let’s start with certificates for some simple facts.
Read Once Certificate for \( \neg \text{PATH} \)

- Let \( C_i \) be the set of vertices reachable from \( s \) in \( G \) within at most \( i \) steps.

- Certificate for the fact that \( v \in C_i \).

The sequence of vertices \( v_0, v_1, \ldots, v_k \) along the path from \( s \) to \( v \) with \( k \leq i \) and \( v_0 = s \).

- The certificate has size \( O(n \log n) \) and can be verified in log space, read once way:
  1. \( v_0 = s \);
  2. \( \forall j > 0, \exists \) edge from \( v_{j-1} \) to \( v_j \);
  3. \( v_k = v \);
  4. \( k \leq i \) (by simple counting)
Read Once Certificate for $\neg$ PATH

- Using the previous certificate, we design two more sophisticated types of certificates.
  
  1. A certificate that a vertex $v \not\in C_i$, assuming the verifier already knows the size of $C_{i-1}$.

  2. A certificate that $|C_i|=c$ for some number $c$, assuming the verifier already knows the size of $C_{i-1}$.
Read Once Certificate for $\neg$ PATH

- A certificate that a vertex $v \not\in C_i$, assuming the verifier already knows the size of $C_{i-1}$.
  
  The list of $|C_{i-1}|$ certificates to the fact that $u \in C_{i-1}$ for every $u \in C_{i-1}$, in ascending order.

- Verifier checks the certificate with size $O(n^2 \log n)$ in log space and read once way:
  
  1. Each certificate is valid.
  2. Any vertex $u$ in the list is indeed larger than the previous vertex (no repeated vertex).
  3. The total number of certificates is exactly $|C_{i-1}|$.
  4. Any vertex $u$ in the list is not $v$ or a neighbor of $v$. 
Read Once Certificate for $\neg$ PATH

- A certificate that $|C_i| = c$ for some number $c$, assuming the verifier already knows the size of $C_{i-1}$.
  
  The list of $n$ certificates for each vertex $u$, either $u \in C_i$ (does not need $|C_{i-1}|$) or $u \not\in C_i$ (needs $|C_{i-1}|$), in ascending order.

- Verifier checks the certificate with size $O(n^3 \log n)$ in log space and read once way:
  
  (1) Each certificate is valid.
  (2) Any vertex $u$ in the list is indeed larger than the previous vertex (no repeated vertex).
  (3) The total number of certificates is exactly $n$.
  (4) The number of vertices $u \in C_i$ is exactly $c$. 
Read Once Certificate for ¬ PATH

- Summarize, we have designed 3 types of certificates:
  1. Certificate for the fact that $v \in C_i$.
  2. Certificate that a vertex $v \not\in C_i$, assuming the verifier already knows the size of $C_{i-1}$.
  3. Certificate that $|C_i|=c$, assuming the verifier already knows the size of $C_{i-1}$.

- The certificate for no path from $s$ to $t$ is equivalent to $t \notin C_n$, and the verifier knows $|C_0|=1$.

Size=$O(n^4 \log n)$
Read Once Certificate for \( \neg \text{PATH} \)

- Implies NL=coNL

- Generalization (using configuration graph): \( \forall \) space constructible function \( S(n) \geq \log n \), NSPACE\((S(n))=\co\text{NSPACE}(S(n))\).

- Summarize: \( L \subseteq \text{NL}=\text{coNL} \subseteq \text{P} \subseteq \text{NP} \subseteq \text{PSPACE}=\text{NPSPACE} \subseteq \text{EXP} \subseteq \text{NEXP} \)