CS760 Topics in Computational Complexity **2024-25 Sem I**

Lecture 11: 03-09-2024

Scribe: Siddharth Patil, Yash Sadhwan *Lecturer: Rohit Gurjar*

1 Outline

In this lecture we will show that

- Reachability is **NL-complete**.
- **CoNL** and **NL** are equal, which is equivalent to showing unreachability $\in$ **NL**.

2 NL-Complete

Definition 11.1. *A language Q is **NL-complete** if every language in **NL** log-space reduces to Q and $Q \in$ **NL**.*

2.1 Log-space Reduction

Definition 11.2. *We say that there is a log-space reduction from a language L to another language L' iff there exists a log-space computable function f such that $\forall x \in L, |f(x)| \leq |x|^c$, where c is a constant, and $x \in L$ iff $f(x) \in L'$.*

2.2 Log-space Computable Functions

The following are two equivalent definitions of a log-space computable function.

Definition 11.3. *A function f is log-space computable iff there exists a Turing Machine M with a read only input tape, a write only output tape and a read/write scratch work tape and the scratch work tape uses only $O(log(n))$ cells, where n is the input size, such that $\forall x\, M(x) = f(x)$.*

Definition 11.4. *A function f is a log-space computable function iff $|f(x)|$ can be computed in log-space and given $i \leq |f(x)|$, the i^{th} bit of $f(x)$ can be computed in log-space.*

3 Reachability is NL-Complete

To prove this, we will first show that every language in **NL** reduces to Reachability and then we will show that Reachabilitily $\in$ **NL**.

3.1 Reachability is NL-Hard

Given any language $Q \in NL$, there exists a log-space NDTM M for Q. Gven the input I for Q, we can construct its configuration graph and number of nodes in this configuration graph will be $O(poly(|I|))$ because the size of the work tape is $O(log(|I|))$, therefore number of work tape configurations is $O(|I|)$ and the number of states is a constant for any given M and the number of possible head positions (on input tape and work tape) is $O(|I| \log |I|)$. Therefore, each vertex in the configuration graph can be represented using $O(log(|I|))$ bits. Also, given two nodes in the configuration graph C and C', we can check whether there is an edge from C to C' in log-space from the transition function δ of M. Hence, given any language $Q \in NL$ and input I, we can compute the corresponding configuration graph in log-space. Now, we pass this graph as the input to Rechability oracle and check if there is a path from the starting configuration to any of the

accepting configurations. If such a path exists, then $I \in Q$, otherwise $I \notin Q$. Therefore, Reachability is **NL-hard**.
NOTE: Here the size of the configuration graph is not $O(log(|I|))$, but instead $O(poly(|I|))$ which is completely fine because our definition of log-space reduction allows the output size to be poly input size.

3.2 Reachability is in NL

It is easy to see that one can construct a NDTM which performs a non deterministic walk on the graph to check the reachability of t from s. In this NDTM, we only need to store the vertex we are currently on, therefore this is a log-space NDTM. Therefore, Rechability is also **NL**.
Therefore, Reachability is **NL-Complete**.

4 Reachability is in CoNL

Theorem 11.5. *(Immerman-Szelepcsényi) If, in some graph G, t is not reachable from s, then there is a certificate for this which is verifiable in log-space (certificate is written on a read-once tape).*

NOTE: The theorem does not state log-space restriction on the size of the certificate and it can be a poly-size certificate. To prove this theorem, we will first have to define **NL** in terms of a verifier DTM.

4.0.1 NL

The following two definitions of **NL** are equivalent.

Definition 11.6. *A language $Q \in$ **NL** iff there is a log-space NDTM M such that $x \in Q$ iff $M(x)$ accepts on some computation path.*

Definition 11.7. *A language $Q \in$ **NL** iff there exists a log-space DTM (verifier) with a read only input tape and an additional read once (only forward) tape (for storing certificate) such that $x \in Q$ iff $\exists y \in \{0,1\}^{|x|^c}$ (certificate) such that $M(x,y) = 1$.*

Now, we have to construct a certificate which should only be read once and still have sufficient information to prove that t is not reachable from s in a given graph, provided that there is no s-t path in the actual graph.
For the construction let us assume that the vertices are numbered, i.e. we can identify distinct vertices using distinct numbers, and let the number of vertices be v. Now, to create the certificate, we will use recurssion. Let us define k_i as the number of distinct nodes reachable from s using paths of length $\leq i$ and let c_i denote the set of distinct nodes reachable from s using paths of length $\leq i$. Then, once we know k_v, our certificate would include the set c_v and the paths from s to these vertices and if t is not one of these vertices, then we can conclude that there is no s-t path in the given graph. Here, the paths are included to verify that all the vertices in c_v are actually reachable from s and are distinct. Also, the list of paths has to be in either ascending or descending order of the vertex numbers reachable from s because the tape being read once, to verify that the final vertex of the given path is different from that of the previous path, we can check the vertex number of this vertex with the one of the previous path and if it is monotonically increasing or decreasing then we can be sure that the k_v paths lead to distinct vertices.
The above certificate will only work if we are sure that the value of k_v is correct therefore we will need to add a certificate for the value of k_v too. This is where recurssion come in. For k_0, the proof is trivial. Now, provided we have a certificate for k_{i-1}, we want to construct a certificate for k_i. To do this, we will verify the inclusion/exclusion of every vertex in the the graph in c_i. To verify the inclusion/exclusion of a given vertex, we will check whether either any of its backwards neighbour or that vertex itself belongs to c_{i-1}, if none of them belong to c_{i-1}, then that is excluded from c_i, otherwise it is included in it. Note that we are not actually storing the list of vertices c_i, but instead, we are only storing its size. Also, to check whether some vertex belongs to c_{i-1} our certificate need to include paths from s to all the vertices in c_{i-1}. This way we can verify the values of k_i. Therefore this certificate can verify no instances of Reachability and hence Reachability is **coNL**.

NOTE: The size of the certificate we constructed above is $O(v^4)$. To verify the inclusion of a vertex in c_{i-1} takes $O(v)$ space (only its path from s is needed). To verify the inclusion/exclusion of a vertex in c_i requires us to do the aforementioned operation at most v times, so the certificate size is $O(v^2)$. The certificate for k_i requires us to repeat the above process v times, therefore its size is $O(n^3)$. And, finally, we need a certificate for all k_i's and there are v of these, therefore the final certificate size comes out to be $O(v^4)$.
NOTE: This also implies that unreachability is in NL and that $coNL = NL$.

5 A stronger result

Immerman and Szelepcsinyi, independently, proved that **NSPACE**$(S(n))$ = **CoNSPACE**$(S(n))$ for all $S(n) \geq log(n)$. The proof is along the same lines as above.