We can construct an algorithm to show that the Reachability problem is in NL:

1. write the index of node $a$ in the work space;
2. if $i$ is the index currently written on the work space:
   2.1 if $i = b$ then accept, else guess an index $j$ (log $n$ bits) and write it on the work space.
   2.2 if $(i, j)$ is not an edge, reject, else replace $i$ by $j$ and return to (2).
Savitch’s Theorem

Further simulation results for nondeterministic space are obtained by other algorithms for Reachability.

We can show that Reachability can be solved by a deterministic algorithm in $O((\log n)^2)$ space.

Consider the following recursive algorithm for determining whether there is a path from $a$ to $b$ of length at most $i$. 
\[ O((\log n)^2) \] space Reachability algorithm:

\[
\text{Path}(a, b, i) \\
\text{if } i = 1 \text{ and } a \neq b \text{ and } (a, b) \text{ is not an edge reject} \\
\text{else if } (a, b) \text{ is an edge or } a = b \text{ accept} \\
\text{else, for each node } x, \text{ check:} \\
1. \text{Path}(a, x, \lfloor i/2 \rfloor) \\
2. \text{Path}(x, b, \lceil i/2 \rceil) \\
\]

if such an \( x \) is found, then accept, else reject.

The maximum depth of recursion is \( \log n \), and the number of bits of information kept at each stage is \( 3 \log n \).
Savitch’s Theorem

The space efficient algorithm for reachability used on the configuration graph of a nondeterministic machine shows:

$$\text{NSPACE}(f) \subseteq \text{SPACE}(f^2)$$

for $$f(n) \geq \log n$$.

This yields

$$\text{PSPACE} = \text{NPSPACE} = \text{co-NPSPACE}.$$
Complementation

A still more clever algorithm for Reachability has been used to show that nondeterministic space classes are closed under complementation:

If \( f(n) \geq \log n \), then

\[
\text{NSPACE}(f) = \text{co-NSPACE}(f)
\]

In particular

\[
\text{NL} = \text{co-NL}.
\]
Logarithmic Space Reductions

We write

\[ A \leq_L B \]

if there is a reduction \( f \) of \( A \) to \( B \) that is computable by a deterministic Turing machine using \( O(\log n) \) workspace (with a read-only input tape and write-only output tape).

*Note:* We can compose \( \leq_L \) reductions. So,

if \( A \leq_L B \) and \( B \leq_L C \) then \( A \leq_L C \)
Analysing carefully the reductions we constructed in our proofs of NP-completeness, we can see that SAT and the various other NP-complete problems are actually complete under $\leq_L$ reductions.

Thus, if $\text{SAT} \leq_L A$ for some problem $A$ in $L$ then not only $P = \text{NP}$ but also $L = \text{NP}$. 