Isolation Lemma for Directed Reachability and NL vs. L

Nutan Limaye (IIT Bombay)
Joint work with Vaibhav Krishan

Dagstuhl Seminar 16411 - Algebraic Methods in Computational Complexity
Oct 09 - Oct 14, 2016
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Valiant Vazirani Isolation Lemma

There is a randomized polynomial time isolation procedure for SAT with success probability $\Omega\left(\frac{1}{n}\right)$. 
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Isolation Procedure for SAT
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula $\phi$ on $n$ input variables,
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula $\phi$ on $n$ input variables, Output: a new formula $\psi$ on the same $n$ variables such that
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula $\phi$ on $n$ input variables,
Output: a new formula $\psi$ on the same $n$ variables such that
- every satisfying assignment of $\psi$ also satisfies $\phi$
Isolation lemma for NP

SAT

Given a Boolean formula \( \phi \) determine whether \( \phi \) has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula \( \phi \) on \( n \) input variables,
Output: a new formula \( \psi \) on the same \( n \) variables such that

- every satisfying assignment of \( \psi \) also satisfies \( \phi \)

This implies that if \( \phi \) is not satisfiable then \( \psi \) is also not satisfiable.
Isolation lemma for NP

\textbf{SAT}

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

\textbf{Isolation Procedure for SAT}

Given: a Boolean formula $\phi$ on $n$ input variables,
Output: a new formula $\psi$ on the same $n$ variables such that

\begin{itemize}
\item every satisfying assignment of $\psi$ also satisfies $\phi$
  
  This implies that if $\phi$ is not satisfiable then $\psi$ is also not satisfiable.
\item if $\phi$ is satisfiable, then $\psi$ has exactly one satisfying assignment.
\end{itemize}
Isolation lemma for NP

SAT

Given a Boolean formula \( \phi \) determine whether \( \phi \) has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula \( \phi \) on \( n \) input variables,
Output: a new formula \( \psi \) on the same \( n \) variables such that

- every satisfying assignment of \( \psi \) also satisfies \( \phi \)
  
  This implies that if \( \phi \) is not satisfiable then \( \psi \) is also not satisfiable.
- if \( \phi \) is satisfiable, then \( \psi \) has exactly one satisfying assignment.

Valiant Vazirani Isolation Lemma
Isolation lemma for NP

SAT

Given a Boolean formula $\phi$ determine whether $\phi$ has a satisfying assignment.

Isolation Procedure for SAT

Given: a Boolean formula $\phi$ on $n$ input variables,
Output: a new formula $\psi$ on the same $n$ variables such that
  ▶ every satisfying assignment of $\psi$ also satisfies $\phi$
    This implies that if $\phi$ is not satisfiable then $\psi$ is also not satisfiable.
  ▶ if $\phi$ is satisfiable, then $\psi$ has exactly one satisfying assignment.

Valiant Vazirani Isolation Lemma

There is a randomized polynomial time isolation procedure for SAT with success probability $\Omega\left(\frac{1}{n}\right)$.
Bumping up the success probability

Determinizing Isolation Procedure for SAT

Can one get rid of the randomness in the Isolation Procedure for SAT?

Bumping up the success probability

Determinizing Isolation Procedure for SAT

Can one get rid of the randomness in the Isolation Procedure for SAT? Open!

Success probability of the Isolation Procedure for SAT

Bumping up the success probability

Determinizing Isolation Procedure for SAT

Can one get rid of the randomness in the Isolation Procedure for SAT? Open!

Success probability of the Isolation Procedure for SAT

Can one make the success probability of the Isolation Procedure higher?

The success probability of the Isolation Procedure for SAT can be made greater than $\frac{2}{3}$ if and only if $\text{NP} \subseteq \text{P/poly}$.

Bumping up the success probability

Determinizing Isolation Procedure for SAT

Can one get rid of the randomness in the Isolation Procedure for SAT? \textit{Open!}

Success probability of the Isolation Procedure for SAT

Can one make the success probability of the Isolation Procedure higher?

The success probability of the Isolation Procedure for SAT can be made greater than $2/3$ if and only if $\text{NP} \subseteq \text{P/poly}$. 
Bumping up the success probability

Determinizing Isolation Procedure for SAT

Can one get rid of the randomness in the Isolation Procedure for SAT? \textbf{Open!}

Success probability of the Isolation Procedure for SAT

Can one make the success probability of the Isolation Procedure higher?

The success probability of the Isolation Procedure for SAT can be made greater than $\frac{2}{3}$ if and only if $\text{NP} \subseteq \text{P/poly}$.

\cite{DKvMW} \textit{Is Valiant Vazinai’s isolation probability improvable?} Dell, Kabanets, van Melkebeek, Watanabe, \textit{Computational Complexity}, 2013.
Directed Reachability,

Given: A directed graph $G = (V, E)$ and two designated vertices $s, t$

Output: yes if and only if there is a directed path from $s$ to $t$ in $G$. 
NL, L, Directed Reachability, L/poly

Complexity classes NL, L, L/poly

- **L**: the class of decision problems decidable by deterministic Turing machines using $O(\log n)$ space, where $n$ is the length of the input.
- **NL**: the class of decision problems decidable by non-deterministic Turing machine using $O(\log n)$ space, where $n$ is the length of the input.
- **L/poly**: the class of decision problems decidable by deterministic Turing machine using $O(\log n)$ space and poly($n$) amount of advice, where $n$ is the length of the input.
NL, L, Directed Reachability, L/poly

Complexity classes NL, L, L/poly

L: the class of decision problems decidable by deterministic Turing machines using $O(\log n)$ space, where $n$ is the length of the input.

NL: the class of decision problems decidable by non-deterministic Turing machine using $O(\log n)$ space, where $n$ is the length of the input.
NL, L, Directed Reachability, L/poly

Complexity classes NL, L, L/poly

L: the class of decision problems decidable by deterministic Turing machines using $O(\log n)$ space, where $n$ is the length of the input.

NL: the class of decision problems decidable by non-deterministic Turing machine using $O(\log n)$ space, where $n$ is the length of the input.

L/poly: the class of decision problems decidable by deterministic Turing machine using $O(\log n)$ space and poly($n$) amount of advice, where $n$ is the length of the input.
Complexity classes NL, L, L/poly

L: the class of decision problems decidable by deterministic Turing machines using $O(\log n)$ space, where $n$ is the length of the input.

NL: the class of decision problems decidable by non-deterministic Turing machine using $O(\log n)$ space, where $n$ is the length of the input.

L/poly: the class of decision problems decidable by deterministic Turing machine using $O(\log n)$ space and $\text{poly}(n)$ amount of advice, where $n$ is the length of the input.

Directed Reachability, Reach

Given: A directed graph $G = (V, E)$ and two designated vertices $s, t$
**NL, L, Directed Reachability, L/poly**

**Complexity classes NL, L, L/poly**

L: the class of decision problems decidable by deterministic Turing machines using $O(\log n)$ space, where $n$ is the length of the input.

NL: the class of decision problems decidable by non-deterministic Turing machine using $O(\log n)$ space, where $n$ is the length of the input.

L/poly: the class of decision problems decidable by deterministic Turing machine using $O(\log n)$ space and poly($n$) amount of advice, where $n$ is the length of the input.

**Directed Reachability, Reach**

Given: A directed graph $G = (V, E)$ and two designated vertices $s, t$
Output: yes if and only if there is a directed path from $s$ to $t$ in $G$. 
Our result

Isolation Procedure for Directed Reachability
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s, t$
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s, t$
Output: a new graph $H$ on the same $n$ variables such that

- every $s$ to $t$ path in $H$ is also a path in $G$
- if $G$ has an $s$ to $t$ path, then $H$ has exactly one $s$ to $t$ path.

Success probability of the Isolation Procedure for Reach

Theorem (Main result)

There exists a randomized isolation procedure for Reach with success probability greater than $\frac{2}{3}$ if and only if $\text{NL} \subseteq \text{L/poly}$. 
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s, t$
Output: a new graph $H$ on the same $n$ variables such that
- every $s$ to $t$ path in $H$ is also a path in $G$
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s, t$
Output: a new graph $H$ on the same $n$ variables such that

- every $s$ to $t$ path in $H$ is also a path in $G$

This implies that in $G$ if $t$ is not reachable from $s$ then in $H$ as well there is no $s$ to $t$ path.
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s, t$
Output: a new graph $H$ on the same $n$ variables such that
- every $s$ to $t$ path in $H$ is also a path in $G$
  This implies that in $G$ if $t$ is not reachable from $s$ then in $H$ as well there is no $s$ to $t$ path.
- if $G$ has an $s$ to $t$ path, then $H$ has exactly one $s$ to $t$ path.
Our result

Isolation Procedure for Directed Reachability

Given: a graph $G$ on $n$ input vertices, and two designated vertices $s$, $t$
Output: a new graph $H$ on the same $n$ variables such that

- every $s$ to $t$ path in $H$ is also a path in $G$
  This implies that in $G$ if $t$ is not reachable from $s$ then in $H$ as well there is no $s$ to $t$ path.
- if $G$ has an $s$ to $t$ path, then $H$ has exactly one $s$ to $t$ path.

Success probability of the Isolation Procedure for Reach

Theorem (Main result)

There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $NL \subseteq L/poly$. 
Isolation for other classes

Isolation for LogCFL

There exists a randomized isolation procedure for a hard problem in LogCFL that runs in L/poly with success probability greater than \( \frac{2}{3} \) if and only if \( \text{LogCFL} \subseteq \text{L/poly} \).

Isolation for NP

There exists a randomized isolation procedure for SAT that runs in L/poly with success probability greater than \( \frac{2}{3} \) if and only if \( \text{NP} \subseteq \text{L/poly} \).
Isolation for other classes

Isolation for LogCFL

There exists a randomized isolation procedure for a hard problem in LogCFL that runs in L/poly with success probability greater than 2/3 if and only if LogCFL ⊆ L/poly.
Isolation for other classes

Isolation for LogCFL

There exists a randomized isolation procedure for a hard problem in LogCFL that runs in $L/poly$ with success probability greater than $2/3$ if and only if $\text{LogCFL} \subseteq L/poly$.

Isolation for NP

There exists a randomized isolation procedure for SAT that runs in $L/poly$ with success probability greater than $2/3$ if and only if $\text{NP} \subseteq L/poly$.
Proof outline

The proofs follow a similar structure as in [DKvMW].
Proof outline

The proofs follow a similar structure as in [DKvMW].
We will start with some definitions.
Proof outline

The proofs follow a similar structure as in [DKvMW]
We will start with some definitions.

Definition (Promise sets for a version of Reach)
Proof outline

The proofs follows a similar structure as in [DKvMW]

We will start with some definitions.

Definition (Promise sets for a version of Reach)

\[ \text{Yes}_{\text{Reach}} = \{(G, s, t) \mid \text{unique reachable path between } s \text{ and } t\}, \]
Proof outline

The proofs follow a similar structure as in [DKvMW].

We will start with some definitions.

**Definition (Promise sets for a version of Reach)**

Yes\(_{\text{Reach}}\) = \{ (G, s, t) | unique reachable path between s and t \}, and

No\(_{\text{Reach}}\) = \{ (G, s, t) | no path between s and t \}
Proof outline

The proofs follows a similar structure as in [DKvMW]

We will start with some definitions.

**Definition (Promise sets for a version of \textbf{Reach})**

\[
\begin{align*}
\text{Yes}_{\text{Reach}} &= \{(G, s, t) \mid \text{unique reachable path between } s \text{ and } t\}, \text{ and} \\
\text{No}_{\text{Reach}} &= \{(G, s, t) \mid \text{no path between } s \text{ and } t\}
\end{align*}
\]

**Definition (\textbf{PrUReach})**
Proof outline

The proofs follows a similar structure as in [DKvMW]

We will start with some definitions.

**Definition (Promise sets for a version of Reach)**

\[
\begin{align*}
\text{Yes}_{\text{Reach}} &= \{(G, s, t) \mid \text{unique reachable path between } s \text{ and } t\}, \text{ and} \\
\text{No}_{\text{Reach}} &= \{(G, s, t) \mid \text{no path between } s \text{ and } t\}
\end{align*}
\]

**Definition (PrUReach)**

Given: \( G \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}} \)
Proof outline

The proofs follow a similar structure as in [DKvMW]

We will start with some definitions.

**Definition (Promise sets for a version of Reach)**

| $\text{Yes}_{\text{Reach}}$ | $\{(G, s, t) \mid$ unique reachable path between $s$ and $t\}$, and |
| $\text{No}_{\text{Reach}}$ | $\{(G, s, t) \mid$ no path between $s$ and $t\}$ |

**Definition (PrUReach)**

Given: $G \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$

Output: yes if and only if $G \in \text{Yes}_{\text{Reach}}$. 
Proof outline

Recall the statement we wish to prove
Proof outline

Recall the statement we wish to prove

Theorem (Main result)

There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $NL \subseteq L/poly$. 
Proof outline

Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for \textbf{Reach} with success probability greater than \( \frac{2}{3} \) if and only if \( \text{NL} \subseteq \text{L/poly} \).

**Step 1** Prove that \( \text{NL} \subseteq \text{L/poly} \) if and only if \( \text{PrUReach} \in \text{L/poly} \).
Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for \textbf{Reach} with success probability greater than $\frac{2}{3}$ if and only if $\text{NL} \subseteq \text{L/poly}$.

**Step 1** Prove that $\text{NL} \subseteq \text{L/poly}$ if and only if $\text{PrUReach} \in \text{L/poly}$.

**Step 2** Prove the following statement:
Proof outline

Recall the statement we wish to prove

Theorem (Main result)

There exists a randomized isolation procedure for \textit{Reach} with success probability greater than $2/3$ if and only if $\text{NL} \subseteq L/\text{poly}$.

Step 1 Prove that $\text{NL} \subseteq L/\text{poly}$ if and only if $\Pr_U \text{Reach} \in L/\text{poly}$.

Step 2 Prove the following statement:
There exists a randomized isolation procedure for \textit{Reach} with success probability greater than $2/3$ if and only if $\Pr_U \text{Reach} \in L/\text{poly}$.
Proof outline

Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $NL \subseteq L/poly$.

**Step 1** Prove that $NL \subseteq L/poly$ if and only if $\Pr_{U, \text{Reach}} \in L/poly$.

**Step 2** Prove the following statement:
There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $\Pr_{U, \text{Reach}} \in L/poly$. 
Details of Step 1

\[ \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \]

This direction is trivial.

Uses an algorithm developed in the following work.

We need one more definition.

Definition (Min-unique graph [GW, RA])
A weighted directed graph \( G = (V, E) \) with weight function \( w : E \to \mathbb{N} \) is said to be min-unique if between every pair of vertices the minimum weight path is unique.
Details of Step 1

\[ \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \]
⇒ This direction is trivial.

Uses an algorithm developed in the following work.

We need one more definition.

**Definition (Min-unique graph)** [GW, RA]

A weighted directed graph \( G = (V, E) \) with weight function \( w : E \to \mathbb{N} \) is said to be **min-unique** if between every pair of vertices the minimum weight path is unique.
Details of Step 1

\[ \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \]

\[ \Rightarrow \] This direction is trivial.

\[ \Leftarrow \] Uses an algorithm developed in the following work.

\[ \text{[RA]} \]
\text{Making Nondeterminism Unambiguous, Reinhardt, Allender, SIAM Journal of Computing, 2000.}
Details of Step 1

\( NL \subseteq L/poly \iff \text{PrUReach} \in L/poly \)

\( \Rightarrow \) This direction is trivial.

\( \Leftarrow \) Uses an algorithm developed in the following work.

Details of Step 1

\[ \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \]

\(\Rightarrow\) This direction is trivial.

\(\Leftarrow\) Uses an algorithm developed in the following work.


We need one more definition.
Details of Step 1

\[ \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \]

⇒ This direction is trivial.

⇐ Uses an algorithm developed in the following work.


We need one more definition.

Definition (Min-unique graph [GW, RA])

Details of Step 1

\( \text{NL} \subseteq \text{L/poly} \iff \text{PrUReach} \in \text{L/poly} \)

\( \Rightarrow \) This direction is trivial.

\( \Leftarrow \) Uses an algorithm developed in the following work.


We need one more definition.

**Definition (Min-unique graph [GW, RA])**

A weighted directed graph \( G = (V, E) \) with weight function \( w : E \rightarrow \mathbb{N} \) is said to be *min-unique* if between every pair of vertices the minimum weight path is unique.
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice

Given a graph $G$ on $n$ vertices a procedure $P_1$ generates graphs $G_1, G_2, \ldots, G_{n^2}$

- For all $1 \leq i \leq n^2$, $G_i$ is on the same set of vertices as $G$. 

$P_1$ has an $L/poly$ algorithm.
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice

Given a graph $G$ on $n$ vertices a procedure $P_1$ generates graphs $G_1, G_2, \ldots, G_{n^2}$

- For all $1 \leq i \leq n^2$, $G_i$ is on the same set of vertices as $G$.

- $G$ has an $s$ to $t$ path iff $\forall i \in [n^2], G_i$ has an $s$ to $t$ path.
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice

Given a graph $G$ on $n$ vertices a procedure $P_1$ generates graphs $G_1, G_2, \ldots, G_{n^2}$

- For all $1 \leq i \leq n^2$, $G_i$ is on the same set of vertices as $G$.

- $G$ has an $s$ to $t$ path iff $\forall i \in [n^2]$, $G_i$ has an $s$ to $t$ path.

- If $G$ has an $s$ to $t$ path then $\exists i \in [n^2]$ : such that $G_i$ is min-unique.
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice

Given a graph $G$ on $n$ vertices a procedure $P_1$ generates graphs $G_1, G_2, \ldots, G_{n^2}$

- For all $1 \leq i \leq n^2$, $G_i$ is on the same set of vertices as $G$.
- $G$ has an $s$ to $t$ path iff $\forall i \in [n^2]$, $G_i$ has an $s$ to $t$ path.
- If $G$ has an $s$ to $t$ path then $\exists i \in [n^2]$ : such that $G_i$ is min-unique.
- $P_1$ has an L/poly algorithm.
Breaking down Step 1

Step 1.1: Generating min-unique graph using advice

Given a graph $G$ on $n$ vertices a procedure $P_1$ generates graphs $G_1, G_2, \ldots, G_{n^2}$

- For all $1 \leq i \leq n^2$, $G_i$ is on the same set of vertices as $G$.

- $G$ has an $s$ to $t$ path iff $\forall i \in [n^2]$, $G_i$ has an $s$ to $t$ path.

- If $G$ has an $s$ to $t$ path then $\exists i \in [n^2]:$ such that $G_i$ is min-unique.

- $P_1$ has an $L/poly$ algorithm.

\[
\begin{array}{c}
\text{advice string} \\
\downarrow \\
G \\
\downarrow \\
P_1 \\
\downarrow \\
\langle G_1, G_2, \ldots, G_{n^2} \rangle
\end{array}
\]
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$
Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$

- On input $(G, s, t)$, $(C_G, s', t')$ is produced.
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$

- On input $(G, s, t)$, $(C_G, s', t')$ is produced.

- $G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$

- On input $(G, s, t)$, $(C_G, s', t')$ is produced.

- $G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

- If $G$ is min-unique and has an $s$ to $t$ path then there is a unique path from $s'$ to $t'$. 
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$:

- On input $(G, s, t)$, $(C_G, s', t')$ is produced.

- $G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

- If $G$ is min-unique and has an $s$ to $t$ path then there is a unique path from $s'$ to $t'$.

- $P_2$ has an L/poly algorithm.
Breaking down Step 1

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices a procedure $P_2$ generates a graph $C_G$

- On input $(G, s, t)$, $(C_G, s', t')$ is produced.

- $G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

- If $G$ is min-unique and has an $s$ to $t$ path then there is a unique path from $s'$ to $t'$.

- $P_2$ has an L/poly algorithm.

![Diagram](diagram.png)
Breaking down Step 1

Step 1.3: Using $P_1$, $P_2$ to solve Reach.
Breaking down Step 1
Step 1.3: Using $P_1$, $P_2$ to solve \textbf{Reach}.

$P$ be the algorithm that solves \textbf{PrUReach} in L/poly.
Breaking down Step 1

Step 1.3: Using $P_1$, $P_2$ to solve Reach.

$P$ be the algorithm that solves \text{PrUR}e\text{ach} in L/poly.

Yes iff $P$ accepts one of the graphs.
Breaking down Step 1

Step 1.3: Using $P_1, P_2$ to solve **Reach**.

$P$ be the algorithm that solves $PrUReach$ in L/poly.

If $G$ does not have an $s$ to $t$ path, we reject.

Yes iff $P$ accepts one of the graphs.
Breaking down Step 1

Step 1.3: Using $P_1$, $P_2$ to solve Reach.

$P$ be the algorithm that solves PrUReach in L/poly.

If $G$ does not have an $s$ to $t$ path, we reject.

If $G$ has an $s$ to $t$ path,
Breaking down Step 1

Step 1.3: Using $P_1$, $P_2$ to solve Reach.

$P$ be the algorithm that solves PrUReach in L/poly.

If $G$ does not have an $s$ to $t$ path, we reject.

If $G$ has an $s$ to $t$ path, at least one of the $G_i$ is min-unique.
Breaking down Step 1

Step 1.3: Using $P_1$, $P_2$ to solve $\text{Reach}$.

Let $P$ be the algorithm that solves $\text{PrUReach}$ in $\text{L/poly}$.

If $G$ does not have an $s$ to $t$ path, we reject.

If $G$ has an $s$ to $t$ path, at least one of the $G_i$ is min-unique.

The corresponding $C_{G_i} \in \text{Yes}_{\text{Reach}}$. 
Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$
Details about Step 1.2

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices

On input $(G, s, t)$, $(C_G, s', t')$ is produced.

$G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.
Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices

On input $(G, s, t)$, $(C_G, s', t')$ is produced.

$G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

If $G$ is min-unique then there is a unique path from $s'$ to $t'$. 
Details about Step 1.2

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices

On input $(G, s, t)$, $(C_G, s', t')$ is produced.

$G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

If $G$ is min-unique then there is a unique path from $s'$ to $t'$.

[RA] If $G$ is min-unqiue then there is a UL algorithm that decides the reachability in $G$. 
Details about Step 1.2

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices

On input $(G, s, t)$, $(C_G, s', t')$ is produced.

$G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

If $G$ is min-unique then there is a unique path from $s'$ to $t'$.

[RA] If $G$ is min-unique then there is a UL algorithm that decides the reachability in $G$.

$C_G$ is the configuration graph of the UL algorithm on input $G$. 
Details about Step 1.2

Step 1.2: Generating $G' \in \text{Yes}_{\text{Reach}} \cup \text{No}_{\text{Reach}}$ given a min-unique $G$

Given a min-unique graph $G$ on $n$ vertices

On input $(G, s, t)$, $(C_G, s', t')$ is produced.

$G$ has an $s$ to $t$ path if and only if $C_G$ has an $s'$ to $t'$ path.

If $G$ is min-unique then there is a unique path from $s'$ to $t'$.

[RA] If $G$ is min-unique then there is a UL algorithm that decides the reachability in $G$.

$C_G$ is the configuration graph of the UL algorithm on input $G$.

$C_G$ can be computed in L.
Proof outline

Recall the statement we wish to prove

Theorem (Main result)

There exists a randomized isolation procedure for \( \text{Reach} \) with success probability greater than \( \frac{2}{3} \) if and only if \( \text{NL} \subseteq \text{L/poly} \).

Step 1 Prove that \( \text{NL} \subseteq \text{L/poly} \) if and only if \( \Pr[U, \text{Reach} \in \text{L/poly}] \).

Step 2 Prove the following statement:

There exists a randomized isolation procedure for \( \text{Reach} \) with success probability greater than \( \frac{2}{3} \) if and only if \( \Pr[U, \text{Reach} \in \text{L/poly}] \).
Proof outline

Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for \textbf{Reach} with success probability greater than \(\frac{2}{3}\) if and only if \(\text{NL} \subseteq \text{L/poly}\).
Proof outline

Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $\text{NL} \subseteq \text{L/poly}$.

**Step 1** Prove that $\text{NL} \subseteq \text{L/poly}$ if and only if $\Pr_{U\text{Reach}} \in \text{L/poly}$. 
## Proof outline

Recall the statement we wish to prove

### Theorem (Main result)

*There exists a randomized isolation procedure for $\text{Reach}$ with success probability greater than $2/3$ if and only if $\text{NL} \subseteq \text{L/poly}$.*

<table>
<thead>
<tr>
<th>Step</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prove that $\text{NL} \subseteq \text{L/poly}$ if and only if $\text{PrUReach} \in \text{L/poly}$.</td>
</tr>
<tr>
<td>2</td>
<td>Prove the following statement:</td>
</tr>
</tbody>
</table>
Proof outline

Recall the statement we wish to prove

Theorem (Main result)

There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $NL \subseteq L/poly$.

Step 1  Prove that $NL \subseteq L/poly$ if and only if $PrUR_{Reach} \in L/poly$.

Step 2  Prove the following statement:
There exists a randomized isolation procedure for Reach with success probability greater than $2/3$ if and only if $PrUR_{Reach} \in L/poly$. 
Proof outline

Recall the statement we wish to prove

**Theorem (Main result)**

There exists a randomized isolation procedure for Reach with success probability greater than $\frac{2}{3}$ if and only if NL $\subseteq$ L/poly.

**Step 1** Prove that NL $\subseteq$ L/poly if and only if $\text{Pr}_U\text{Reach} \in L/\text{poly}$.

**Step 2** Prove the following statement:

There exists a randomized isolation procedure for Reach with success probability greater than $\frac{2}{3}$ if and only if $\text{Pr}_U\text{Reach} \in L/\text{poly}$. 
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say $B$ s.t.
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say $B$ s.t.

Given a graph $G$ as input, it outputs $\langle G_1, G_2, \ldots, G_t \rangle$ such that $> 2/3$ fraction of the $G_i$s have unique $s$ to $t$ paths.
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say $B$ s.t.

Given a graph $G$ as input, it outputs $\langle G_1, G_2, \ldots, G_t \rangle$ such that $> \frac{2}{3}$ fraction of the $G_i$s have unique $s$ to $t$ paths.
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say $B$ s.t.

Given a graph $G$ as input, it outputs $\langle G_1, G_2, \ldots, G_t \rangle$ such that $> \frac{2}{3}$ fraction of the $G_i$s have unique $s$ to $t$ paths.

$H$ be a graph with $\pi$ as its $s$ to $t$ path.
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say \( B \) s.t.

Given a graph \( G \) as input, it outputs \( \langle G_1, G_2, \ldots, G_t \rangle \) such that > 2/3 fraction of the \( G_i \)s have unique \( s \) to \( t \) paths.

\[ \text{advice string} \]

\[ G \rightarrow B \rightarrow \langle G_1, G_2, \ldots, G_t \rangle \]

\( H \) be a graph with \( \pi \) as its \( s \) to \( t \) path.

For two graph \( G, H \) we say that \( (G, H) \) is good if \( \pi \) is a reachable path in < 2/3 fraction of graphs in \( B(G + H) \).
Details about Step 2

From the hypothesis we can show that there is an L/poly procedure, say \( B \) s.t.

Given a graph \( G \) as input, it outputs \( \langle G_1, G_2, \ldots, G_t \rangle \) such that > 2/3 fraction of the \( G_i \)'s have unique \( s \) to \( t \) paths.

\[
\text{advice string}
\]

\[
G \quad \xrightarrow{} \quad B \quad \xrightarrow{} \quad \langle G_1, G_2, \ldots, G_t \rangle
\]

\( H \) be a graph with \( \pi \) as its \( s \) to \( t \) path.

For two graph \( G, H \) we say that \( (G, H) \) is good if \( \pi \) is a reachable path in < 2/3 fraction of graphs in \( B(G + H) \).

Let \( B(G + H) = \langle G_1, G_2, \ldots, G_t \rangle \).
Details about Step 2

Properties of a good pair \((G, H)\)
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\)

Let \(\pi, \rho\) be unique s to t paths in \(H, G\), respectively. \((\pi \neq \rho)\).

If neither good, then each \(\pi\) and \(\rho\) are reachable paths in \(2/3\) of the \(G_i\)s.

This means > \(1/3\) of \(G_i\)s have two distinct s to t paths.

This contradicts the hypothesis of \(B\).

Given \(H, \pi\) as advice and \(G\) as input, whether \((G, H)\) is good or not can be decided in \(L/\text{poly}\).
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\).
Details about Step 2

Properties of a good pair $(G, H)$

If $(G, H)$ is good then $G \in \text{Yes}_{\text{Reach}}$.

If $(G, H)$ good then there is a $\rho$ such that $\rho \neq \pi$ and $\rho$ is a unique reachable path in some $G_i$, that $\rho$ is a reachable path in $G$. 
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively.
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively. \((\pi \neq \rho)\).
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively. \((\pi \neq \rho)\).

If neither good, then each \(\pi\) and \(\rho\) are reachable paths in \(2/3\) of the \(G_i\)s.
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively. \((\pi \neq \rho)\).

If neither good, then each \(\pi\) and \(\rho\) are reachable paths in \(2/3\) of the \(G_i\)s.

This means > \(1/3\) of \(G_i\)s have two distinct \(s\) to \(t\) paths.
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively. \((\pi \neq \rho)\).

If neither good, then each \(\pi\) and \(\rho\) are reachable paths in \(2/3\) of the \(G_i\)s.

This means \(> 1/3\) of \(G_i\)s have two distinct \(s\) to \(t\) paths.
This contradicts the hypothesis of \(B\).
Details about Step 2

Properties of a good pair \((G, H)\)

If \((G, H)\) is good then \(G \in \text{Yes}_{\text{Reach}}\).

If \((G, H)\) good then there is a \(\rho\) such that \(\rho \neq \pi\) and \(\rho\) is a unique reachable path in some \(G_i\), that \(\rho\) is a reachable path in \(G\). Therefore \(G \in \text{Yes}_{\text{Reach}}\).

If \(G, H\) are both in \(\text{Yes}_{\text{Reach}}\) then either \((G, H)\) or \((H, G)\) is good.

Let \(\pi, \rho\) be unique \(s\) to \(t\) paths in \(H, G\), respectively. \((\pi \neq \rho)\).

If neither good, then each \(\pi\) and \(\rho\) are reachable paths in \(2/3\) of the \(G_i\)s.

This means \(> 1/3\) of \(G_i\)s have two distinct \(s\) to \(t\) paths. This contradicts the hypothesis of \(B\).

Given \(H, \pi\) as advice and \(G\) as input, whether \((G, H)\) is good or not can be decided in \(L/poly\).
Wrap-up

Design the advice strings

As advice we need \((H_1, \pi_1), (H_2, \pi_2), \ldots, (H_\ell, \pi_\ell)\).
Wrap-up

Design the advice strings

As advice we need $(H_1, \pi_1), (H_2, \pi_2), \ldots, (H_\ell, \pi_\ell)$.

The advice ensures that
Wrap-up

Design the advice strings

As advice we need \((H_1, \pi_1), (H_2, \pi_2), \ldots, (H_\ell, \pi_\ell)\).

The advice ensures that if \(G \in \text{Yes}_{\text{Reach}}\) then there is an \(H_i\) such that \(H_i \in \text{Yes}_{\text{Reach}}\) and \(\pi_i\) is corresponding path.
Design the advice strings

As advice we need \((H_1, \pi_1), (H_2, \pi_2), \ldots, (H_\ell, \pi_\ell)\).

The advice ensures that if \(G \in \text{Yes}_{\text{Reach}}\) then there is an \(H_i\) such that \(H_i \in \text{Yes}_{\text{Reach}}\) and \(\pi_i\) is corresponding path.

If \(G \in \text{No}_{\text{Reach}}\), then each \(H_i \in \text{No}_{\text{Reach}}\).
Wrap-up

Design the advice strings

As advice we need \((H_1, \pi_1), (H_2, \pi_2), \ldots, (H_\ell, \pi_\ell)\).

The advice ensures that if \(G \in \text{Yes}_{\text{Reach}}\) then there is an \(H_i\) such that \(H_i \in \text{Yes}_{\text{Reach}}\) and \(\pi_i\) is corresponding path.

If \(G \in \text{No}_{\text{Reach}}\), then each \(H_i \in \text{No}_{\text{Reach}}\).

Putting it together

Overall, this gives a L/poly algorithm for \(\text{PrUReach}\).
Thank You!