# Myhill-Nerode Theorem

Deepak D'Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

30 January 2024

# Outline

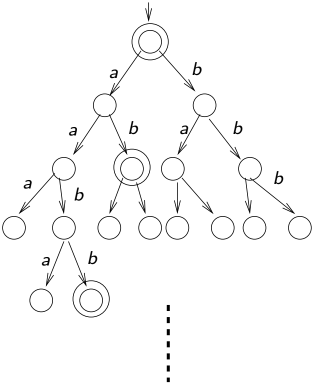
# Myhill-Nerode Theorem: Overview

- A language $L$ is regular iff a certain equivalence relation induced by $L$ (called $\equiv_L$) has a finite number of equivalence classes.
- Every language $L$ has a "canonical" deterministic automaton accepting it.
  - Every other DA for $L$ is a "refinement" of this canonical DA.
  - There is a unique DA for $L$ with the minimal number of states.
- Holds for any $L$ (not just regular $L$).
- $L$ is regular iff this canonical DA has a finite number of states.
- There is an algorithm to compute this canonical DA from any given finite-state DA for $L$.

# DA for any language

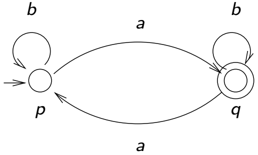
Note that every language $L$ has DA accepting it (we call this the "free" DA for $L$).

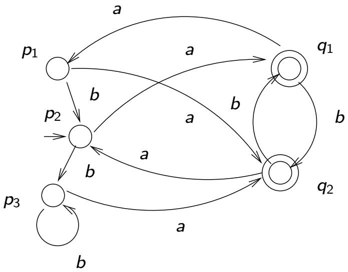
The free DA for $L = \{a^n b^n \mid n \geq 0\}$:



# Illustrating "refinement" of DA: Example 0

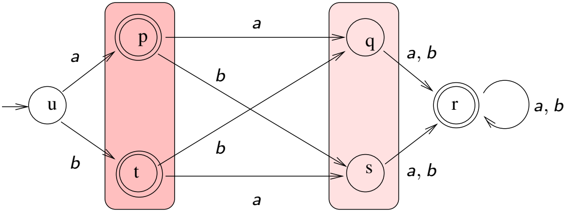
- Replicate each state $p$ in the first automaton some number of times ($p_1$, $p_2$, ...), and add an edge labelled $a$ from each $p_i$ to some $q_j$ such that $\delta(p, a) = q$. The "split" DA accepts the same language.
- Conversely, every DA for $L$ is a "splitting" of the canonical DA for $L$.

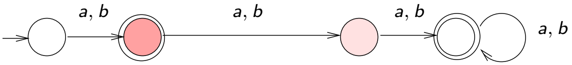


# Illustrating "refinement" of DA: Example 1

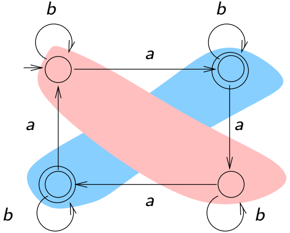
Every DA for $L$ is a "refinement" of this canonical DA:

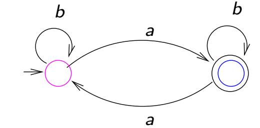


# Illustrating "refinement" of DA: Example 2

Every DA for $L$ is a "refinement" of this canonical DA:



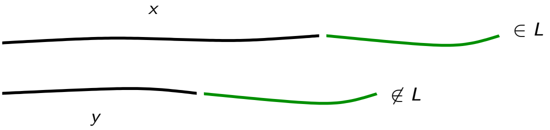



# Myhill-Nerode Theorem

Canonical equivalence relation $\equiv_L$ on $A^*$ induced by $L \subseteq A^*$:

$$x \equiv_L y \text{ iff } \forall z \in A^*,\ xz \in L \text{ iff } yz \in L.$$

$x \not\equiv_L y$ iff




**Theorem (Myhill-Nerode)**

*$L$ is regular iff $\equiv_L$ is of finite index (that is has a finite number of equivalence classes).*

## Exercise 1

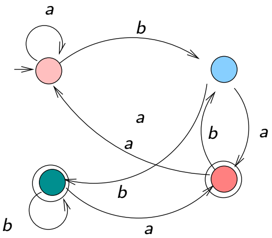
Describe the equivalence classes for $L =$ "Odd number of $a$'s".

# Exercise 2

Describe precisely the equivalence classes of $\equiv_L$ for the language $L \subseteq \{a, b\}^*$ comprising strings in which the 2nd last letter is a $b$.

# Exercise 2

Describe precisely the equivalence classes of $\equiv_L$ for the language $L \subseteq \{a, b\}^*$ comprising strings in which the 2nd last letter is a $b$.



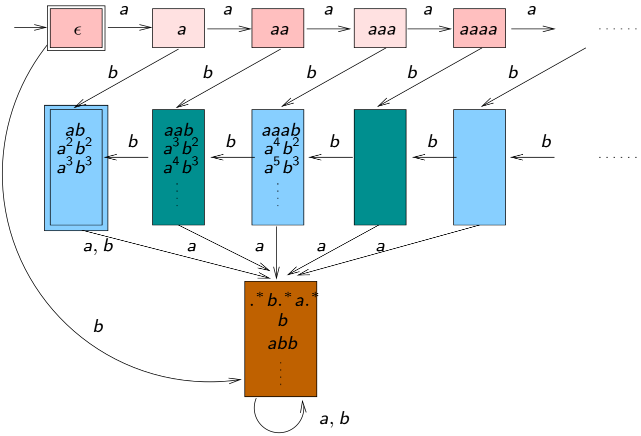

| | |
|---|---|
| $\epsilon$, $a$, $.*aa$ | $b$, $.*ab$ |
| $.*bb$ | $.*ba$ |

# Exercise 3

Describe the equivalence classes of $\equiv_L$ for the language $L = \{a^n b^n \mid n \geq 0\}$.

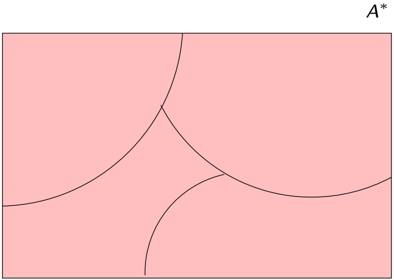
# Exercise 3

Describe the equivalence classes of $\equiv_L$ for the language $L = \{a^n b^n \mid n \geq 0\}$.



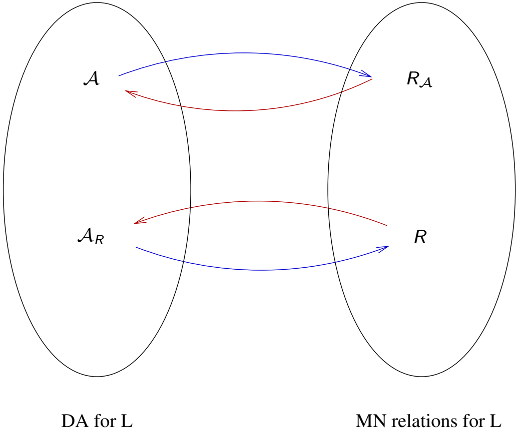
# Myhill-Nerode (MN) relations for a language

- An MN relation for a language $L$ on an alphabet $A$ is an equivalence relation $R$ on $A^*$ satisfying
  1. $R$ is right-invariant (i.e. $xRy \implies xaRya$ for each $a \in A$.)
  2. $R$ refines (or "respects") $L$ (i.e. $xRy \implies x, y \in L \text{ or } x, y \notin L$).



# Deterministic Automata for $L$ and MN relations for $L$

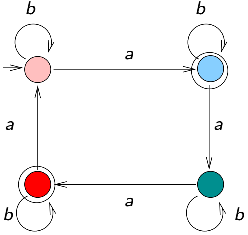
DA for $L$ and MN relations for $L$ are in 1-1 correspondence (they represent eachother).



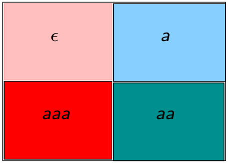

Maps $\mathcal{A} \mapsto R_{\mathcal{A}}$ and $\mathcal{A}_R \leftarrow\!\shortmid R$ are inverses of eachother.

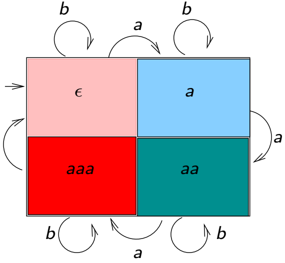
# Example DA and its induced MN relation

$L$ is "Odd number of $a$'s":




$\mathcal{A} \mapsto R_{\mathcal{A}}$

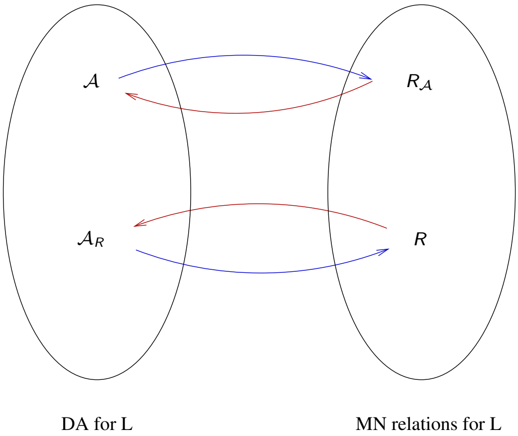



$R \mapsto \mathcal{A}_R$

# Deterministic Automata for $L$ and MN relations for $L$

DA (with no unreachable states) for $L$ and MN relations for $L$ are in 1-1 correspondence.




Maps $\mathcal{A} \mapsto R_{\mathcal{A}}$ and $\mathcal{A}_R \leftarrow\!\shortmid R$ are inverses of eachother.

# Equivalence relations and Refinement

An equivalence relation $R$ on a set $X$ refines another equivalence relation $S$ on $X$ if for each $x, y \in X$, $xRy \implies xSy$.

Exercise: Consider the relations $R$: "equal mod 2" and $S$: "equal mod 4" on $\mathbb{N}$. Which refines which? Picture $R$ and $S$.

# Any MN-relation for $L$ refines the relation $\equiv_L$

**Lemma**

*Let $L$ be any language over an alphabet $A$. Let $R$ be any MN-relation for $L$. Then $R$ refines $\equiv_L$.*

# Any MN-relation for $L$ refines the relation $\equiv_L$

**Lemma**

*Let $L$ be any language over an alphabet $A$. Let $R$ be any MN-relation for $L$. Then $R$ refines $\equiv_L$.*

Proof: To prove that $xRy$ implies $x \equiv_L y$. Suppose $x \not\equiv_L y$. Then there exists $z$ such that (WLOG) $xz \in L$ and $yz \notin L$. Suppose $xRy$. Since its an MN relation for $L$, it must be right invariant; and hence $xzRyz$. But this contradicts the assumption that $R$ respects $L$.

## Any MN-relation for $L$ refines the relation $\equiv_L$

**Lemma**

*Let $L$ be any language over an alphabet $A$. Let $R$ be any MN-relation for $L$. Then $R$ refines $\equiv_L$.*

Proof: To prove that $xRy$ implies $x \equiv_L y$. Suppose $x \not\equiv_L y$. Then there exists $z$ such that (WLOG) $xz \in L$ and $yz \notin L$. Suppose $xRy$. Since its an MN relation for $L$, it must be right invariant; and hence $xzRyz$. But this contradicts the assumption that $R$ respects $L$.

As a corollary we have:

**Theorem (Myhill-Nerode)**

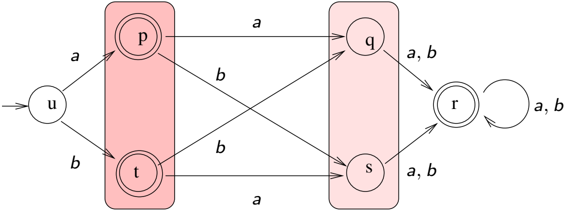
*$L$ is regular iff $\equiv_L$ is of finite index (that is has a finite number of equivalence classes).*

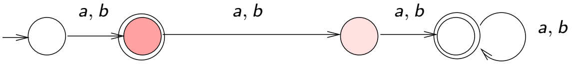
# Canonical DA for $L$

- We call $\mathcal{A}_{\equiv_L}$ the "canonical" DA for $L$.
- In what sense is $\mathcal{A}_{\equiv_L}$ canonical?
  - Every other DA for $L$ is a refinement of $\mathcal{A}_{\equiv_L}$.
  - $\mathcal{A}$ is a refinement of $\mathcal{B}$ if there is a stable partitioning $\sim$ of $\mathcal{A}$ such that quotient of $\mathcal{A}$ under $\sim$ (written $\mathcal{A}/\sim$) is isomorphic to $\mathcal{B}$.
  - Stable partitioning of $\mathcal{A} = (Q, s, \delta, F)$ is an equivalence relation $\sim$ on $Q$ such that:
    - $p \sim q$ implies $\delta(p, a) \sim \delta(q, a)$.
    - If $p \sim q$ and $p \in F$, then $q \in F$ also.
  - Note that if $\sim$ is a stable partitioning of $\mathcal{A}$, then $\mathcal{A}/\sim$ accepts the same language as $\mathcal{A}$.

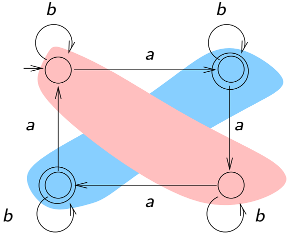
# Example: 1

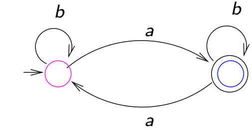
A stable partitioning shown by pink and light pink classes, and below, the quotiented automaton:



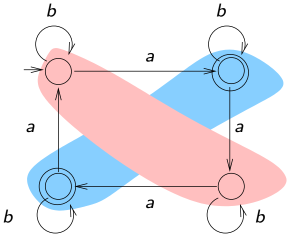
# Example: 2



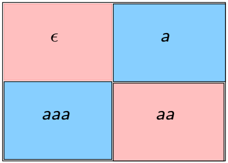



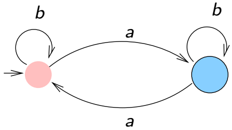
# Proving canonicity of $\mathcal{A}_{\equiv_L}$

Let $\mathcal{A}$ be a DA for $L$ with no unreachable states. Then $\mathcal{A}_{\equiv_L}$ represents a stable partitioning of $\mathcal{A}$. (Use the refinement of $\equiv_L$ by the MN relation $R_{\mathcal{A}}$.)

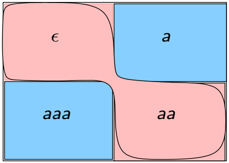



$\mathcal{A} \mapsto R_{\mathcal{A}}$




$\mathcal{A}_{\equiv_L} \leftarrow\!\dashv \equiv_L$



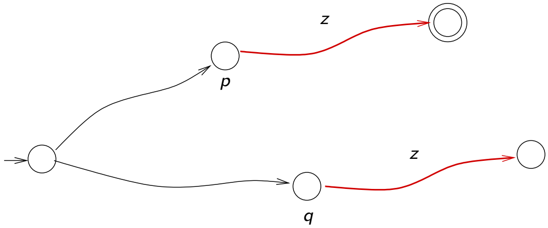
## Stable partitioning $\approx$

- Let $\mathcal{A} = (Q, s, \delta, F)$ be a DA for $L$ with no unreach. states.
- The canonical MN relation for $L$ (i.e. $\equiv_L$) induces a "coarsest" stable partitioning $\approx_L$ of $\mathcal{A}$ given by

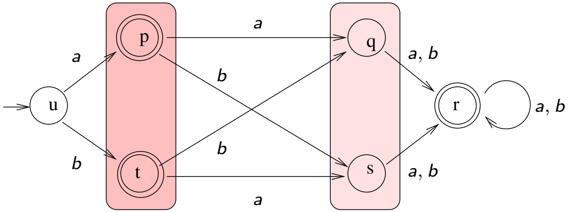
$$p \approx_L q \quad \text{iff} \quad \exists x, y \in A^* \text{ such that } \widehat{\delta}(s, x) = p \text{ and } \widehat{\delta}(s, y) = q, \text{ with } x \equiv_L y.$$

- Define a stable partitioning $\approx$ of $\mathcal{A}$ by

$$p \approx q \text{ iff } \forall z \in A^* : \ \widehat{\delta}(p, z) \in F \text{ iff } \widehat{\delta}(q, z) \in F.$$



# Example of $\approx$ partitioning relation



# Stable partitioning $\approx$ is coarsest

**Claim: $\approx$ coincides with $\approx_L$.**

$\approx_L = \approx$.

Proof:

$$
\begin{aligned}
p \not\approx q \text{ iff } & \exists x, y, z : \ \widehat{\delta}(s, x) = p,\ \widehat{\delta}(s, y) = q, \text{ and} \\
& \quad \widehat{\delta}(p, z) \in F \text{ but } \widehat{\delta}(q, z) \notin F. \\
\text{iff } & p \not\approx_L q.
\end{aligned}
$$

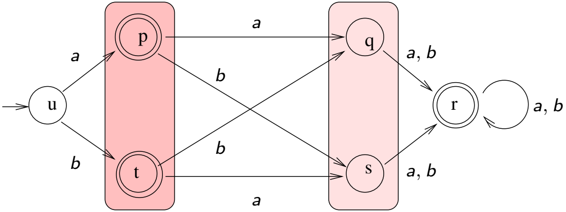
# Algorithm to compute $\approx$ for a given DFA

Input: DFA $\mathcal{A} = (Q, s, \delta, F)$.
Output: $\approx$ for $\mathcal{A}$.

1. Create a (symmetric) table indexed by pairs of states. Initialize entry for each pair to "unmarked".
2. Mark $(p, q)$ if $p \in F$ and $q \notin F$ (or vice-versa).
3. Call a pair $(p, q)$ markable if $(p, q)$ is unmarked, and for some $a \in A$, the pair $(\delta(p, a), \delta(q, a))$ is marked.
4. While there is an markable pair:
    1. Pick a markable pair $(p, q)$ and mark it.
5. Return $\approx$ as: $p \approx q$ iff $(p, q)$ is left unmarked in table.

# Example

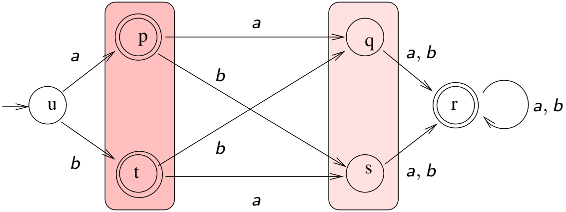
Run minimization algorithm on DFA below:




| | $u$ | $p$ | $t$ | $q$ | $s$ | $r$ |
|---|---|---|---|---|---|---|
| $u$ | · | | | | | |
| $p$ | | · | | | | |
| $t$ | | | · | | | |
| $q$ | | | | · | | |
| $s$ | | | | | · | |
| $r$ | | | | | | · |

# Example

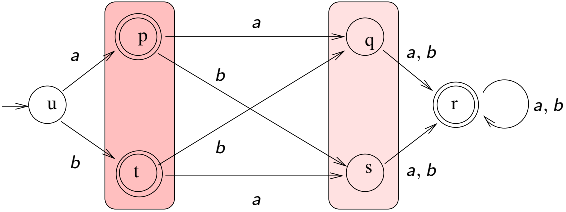
Run minimization algorithm on DFA below:




| | $u$ | $p$ | $t$ | $q$ | $s$ | $r$ |
|---|---|---|---|---|---|---|
| $u$ | · | | | | | |
| $p$ | $\surd$ | · | | | | |
| $t$ | $\surd$ | | · | | | |
| $q$ | | $\surd$ | $\surd$ | · | | |
| $s$ | | $\surd$ | $\surd$ | | · | |
| $r$ | $\surd$ | | | $\surd$ | $\surd$ | · |

# Example

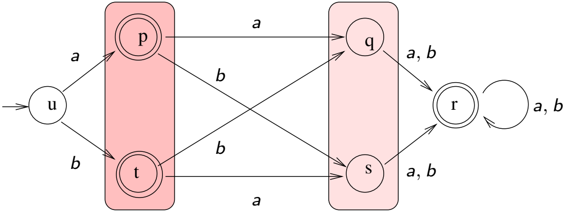
Run minimization algorithm on DFA below:




| | $u$ | $p$ | $t$ | $q$ | $s$ | $r$ |
|---|---|---|---|---|---|---|
| $u$ | · | | | | | |
| $p$ | √ | · | | | | |
| $t$ | √ | | · | | | |
| $q$ | | √ | √ | · | | |
| $s$ | | √ | √ | | · | |
| $r$ | √ | √ | √ | √ | √ | · |

# Example

Run minimization algorithm on DFA below:

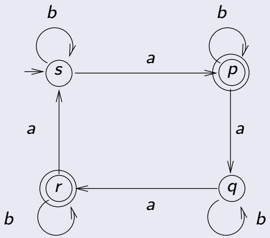



| | $u$ | $p$ | $t$ | $q$ | $s$ | $r$ |
|---|---|---|---|---|---|---|
| $u$ | · | | | | | |
| $p$ | $\surd$ | · | | | | |
| $t$ | $\surd$ | | · | | | |
| $q$ | $\surd$ | $\surd$ | $\surd$ | · | | |
| $s$ | $\surd$ | $\surd$ | $\surd$ | | · | |
| $r$ | $\surd$ | $\surd$ | $\surd$ | $\surd$ | $\surd$ | · |

# Exercise

## Exercise

Run Algorithm to compute $\approx$ for DFA below:



# Correctness of minimization algorithm

Claim: Algo always terminates. Let $n = |Q|$.

- $n(n-1)/2$ table entries in each scan, and at most $n(n-1)/2$ scans.
- In fact, number of scans in algo is $\leq n$.
  1. Consider modified Step 3.1 in which mark check is done wrt the table at the end of previous scan.
  2. Argue that at end of $i$-th scan algo computes $\approx_i$, where

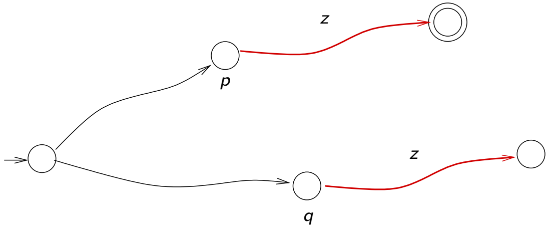
     $$p \approx_i q \text{ iff } \forall w \in A^* \text{ with } |w| \leq i : \ \widehat{\delta}(p, w) \in F \text{ iff } \widehat{\delta}(q, w) \in F.$$

  3. Observe that $\approx_{i+1}$ strictly refines $\approx_i$, unless the algo terminates after scan $i+1$. So modified algo does at most $n$ scans.
  4. Both versions mark the same set of pairs. Also if modified algo marks a pair, original algo has already marked it.
- Hence algo runs in $O(n^3)$ time.

# Correctness of minimization algorithm

Claim: Original algo marks $(p, q)$ iff $p \not\approx q$.

- ($\Rightarrow$:) Argue by induction on number of steps of the algo that this is true.
- ($\Leftarrow$:) Suppose $p \not\approx q$. Argue by induction on $n$ that if two states $t$ and $u$ are distinguished by a string $z$ of length $n$, then $(t, u)$ will be marked by the algo.



# A corollary

If $p$ and $q$ are two states such that $p \not\approx q$, then there is a string of length at most $n-2$ which distinguishes them.