CSE 191

Introduction to Discrete Structures

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Finite Automata and Regular Expressions

Outline

- **Computing with Limited Resources**
  - **Finite Automata (Model of Computation)**
  - Regular Expressions

Computing with Limited Resources

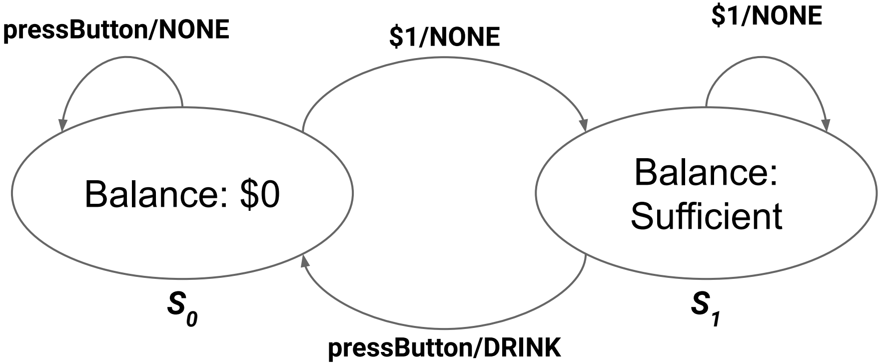
Example

Design the payment accepting component of a vending machine

Computing with Limited Resources

Assume

- Price of a drink/snack is $1
- Denominations accepted: $1 dollar bills
- No change is given
- Enter $1, press a button, machine gives you a drink/snack



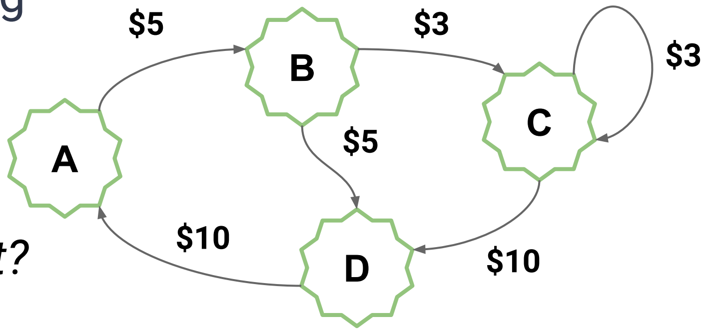
State Table for a Vending Machine				
	Next State		Output	
	Input		Input	
	$1	pressButton	$1	pressButton
S_0	S_1	S_0	NONE	NONE
S_1	S_1	S_0	NONE	DRINK

Computing with Limited Resources

Example

Consider a park with a group of islands connected by 1-way bridges

- Each bridge has a toll to pay for each crossing
- Park entrance: ***A***, Park exit: ***D***



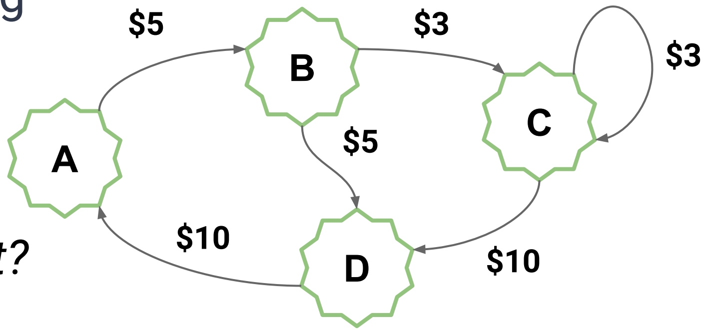
What is a valid fare sequence from entrance to exit?

Computing with Limited Resources

<u>**Example**</u>

Consider a park with a group of islands connected by 1-way bridges

- Each bridge has a toll to pay for each crossing
- Park entrance: ***A***, Park exit: ***D***



What is a valid fare sequence from entrance to exit?

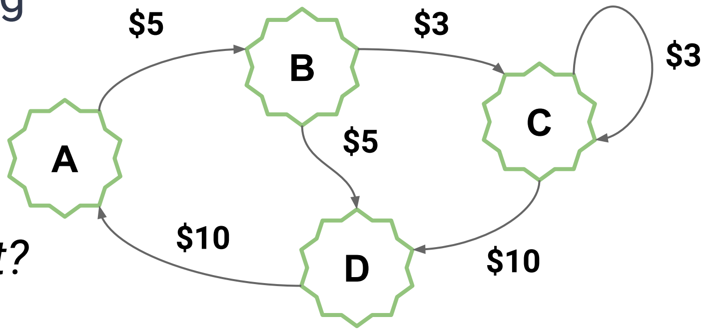
- The shortest drive: ***A, B, D***
    - ***A*** → ***B***: pay $5 ***B*** → ***D***: pay $5
    - Fare sequence: $5, $5

Computing with Limited Resources

<u>Example</u>

Consider a park with a group of islands connected by 1-way bridges

- Each bridge has a toll to pay for each crossing
- Park entrance: ***A***, Park exit: ***D***



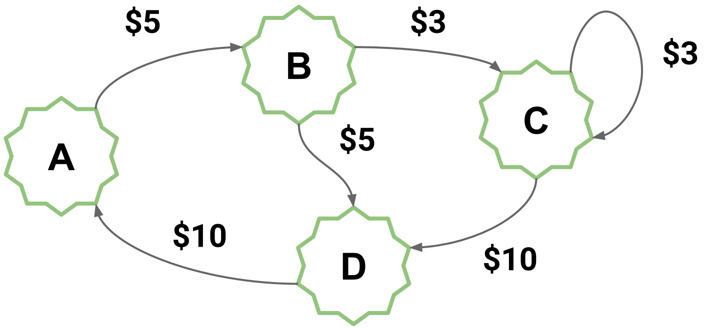
What is a valid fare sequence from entrance to exit?

- Another drive: ***A***, ***B***, ***C***, ***C***, ***D***, ***A***, ***B***, ***C***, ***D***
  - ***A*** → ***B***: pay \$5 ***B*** → ***C***: pay \$3 … ***C*** → ***D***: pay \$10
  - Fare sequence: \$5, \$3, \$3, \$10, \$10, \$5, \$3, \$10

Computing with Limited Resources

*Does the fare sequence $5, $5, $10, $5, $3 end at **D**?*

What about $5, $10, $10?

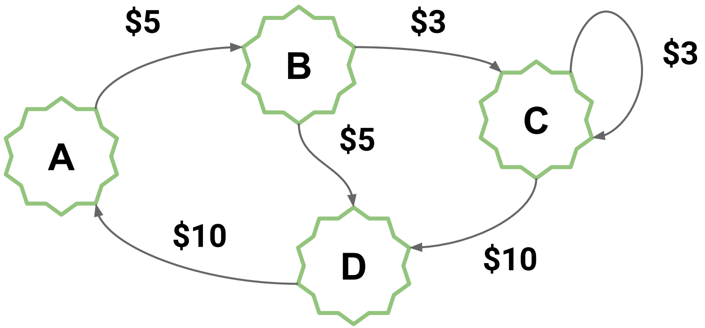


Computing with Limited Resources

*Does the fare sequence \$5, \$5, \$10, \$5, \$3 end at **D**?*

No: *A, B, D, A, B, C*

What about \$5, \$10, \$10?



Computing with Limited Resources

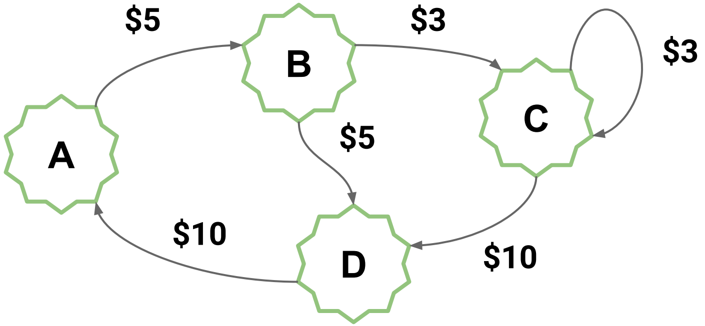
*Does the fare sequence \$5, \$5, \$10, \$5, \$3 end at **D**?*

No: *A, B, D, A, B, C*

What about \$5, \$10, \$10?

No: *A, B, ???*

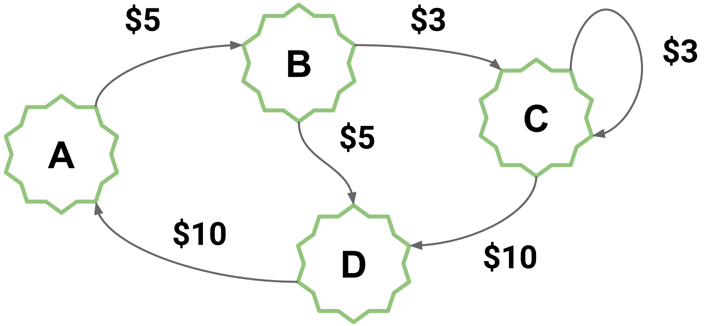
Not even a valid sequence



Computing with Limited Resources

Suppose we survey visitors of the park on what route they took

How can we validate their response?

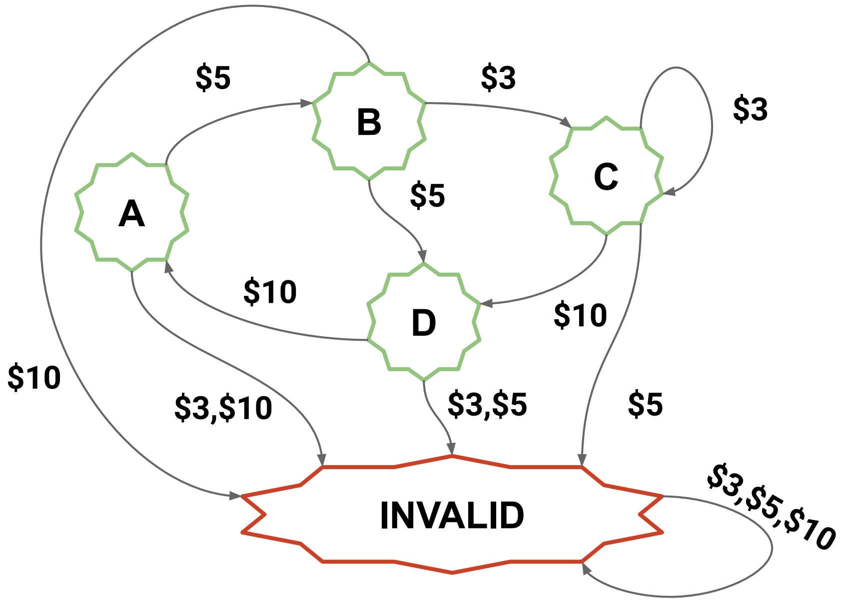


Computing with Limited Resources

Suppose we survey visitors of the park on what route they took

How can we validate their response?

Account for INVALID scenarios



Computing with Limited Resources

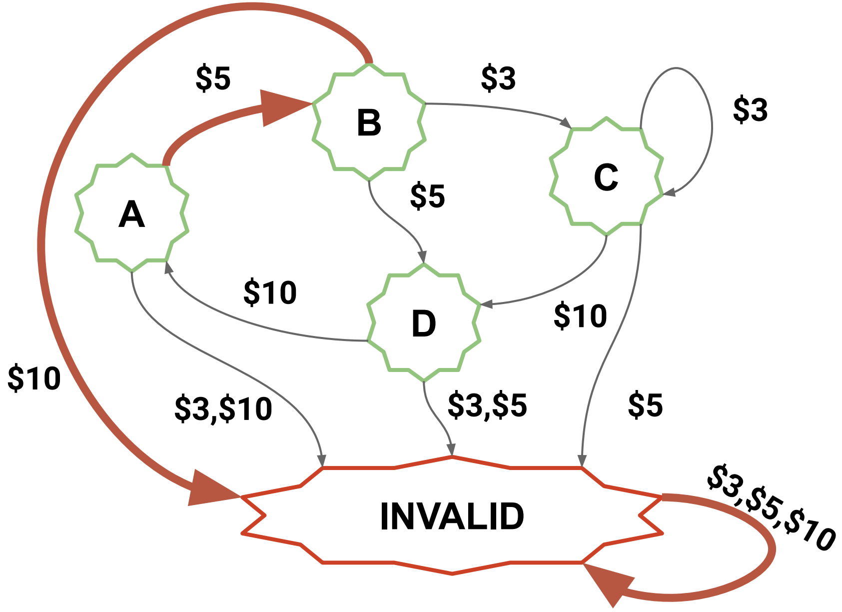
Suppose we survey visitors of the park on what route they took

How can we validate their response?

Account for INVALID scenarios

Reconsider the sequence \$5, \$10, \$10

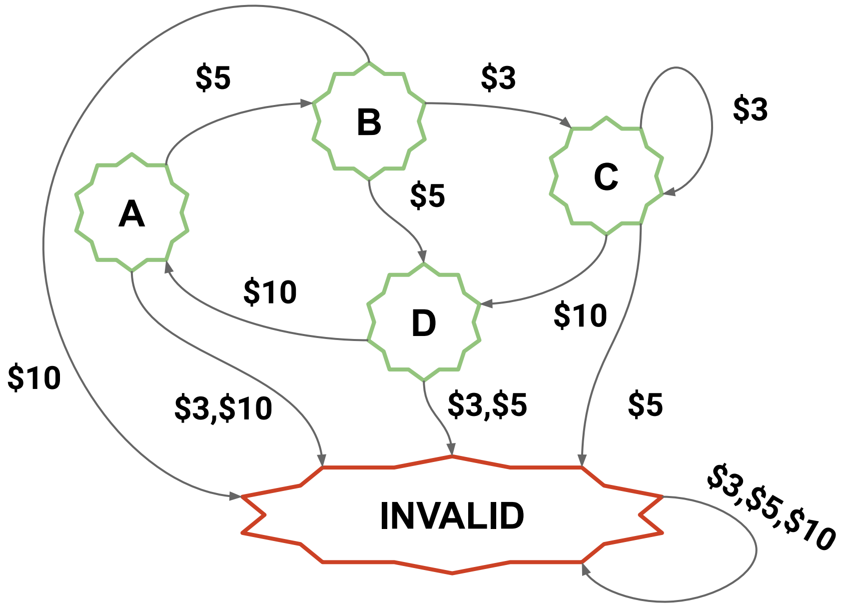
- ***A***, ***B***, **INVALID, INVALID**



Computing with Limited Resources

What is needed to create this program?

- **Vertices**: states of computation
- **Edges**: state transitions
- **Edge labels**: symbols we process
- **Start**: where to start out computation
- **Final**: where it is OK/valid to end



Finite Automata - Finite State Machines (with no output)

A (deterministic) **finite automaton** ***M*** is a 5-tuple (***Q*, Σ, δ, *q*$_0$, *F***) where:

- The **set of states *Q*** is finite and non-empty
- The **input alphabet Σ** is finite and non-empty
- The **transition function δ: $Q \times \Sigma \rightarrow Q$**
- The **starting state $q_0 \in Q$**
- The **set of final states *F***

Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

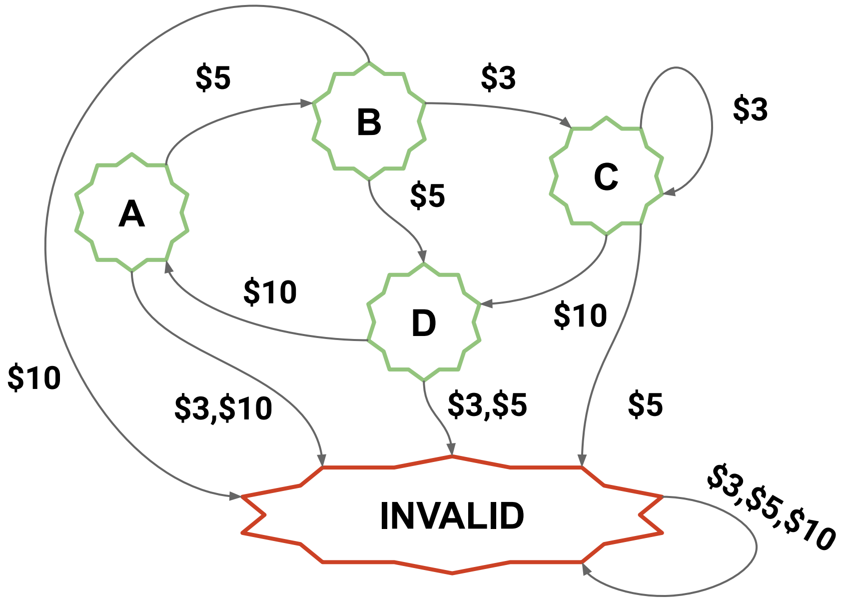
$Q_1 =$

$\Sigma_1 =$

$S_1 =$

$F_1 =$

$\delta_1 =$



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

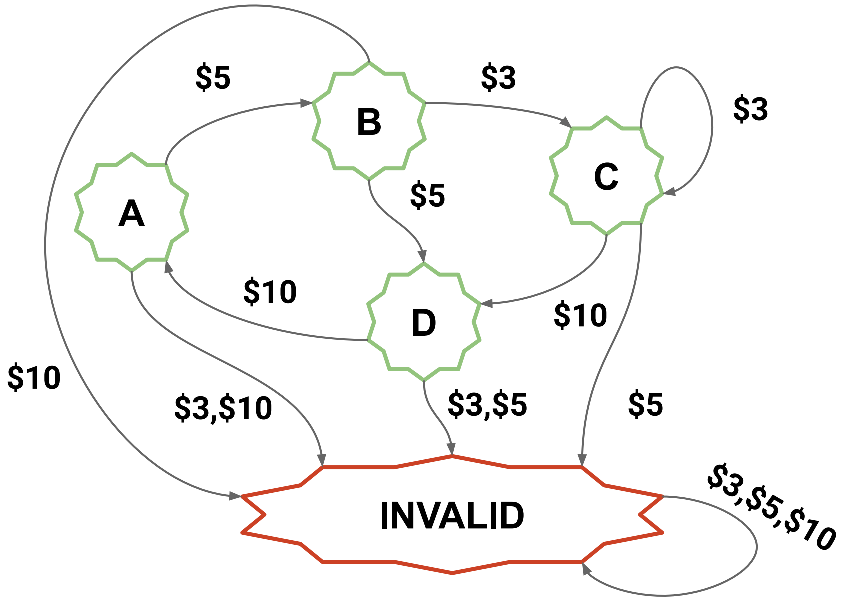
Q_1 = { *A*, *B*, *C*, *D*, INVALID }

Σ_1 =

S_1 =

F_1 =

δ_1 =



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

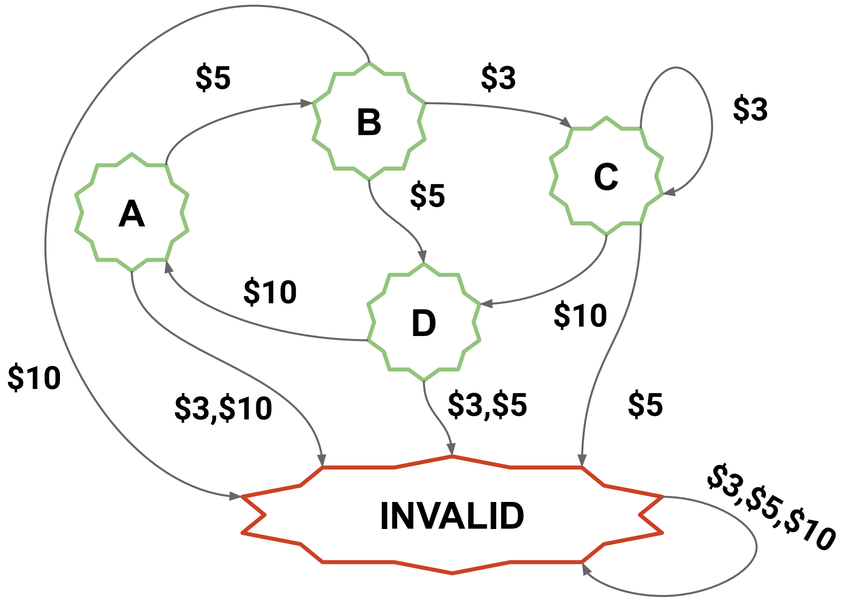
Q_1 = { A, B, C, D, INVALID }

Σ_1 = { \$3, \$5, \$10 }

S_1 =

F_1 =

δ_1 =



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

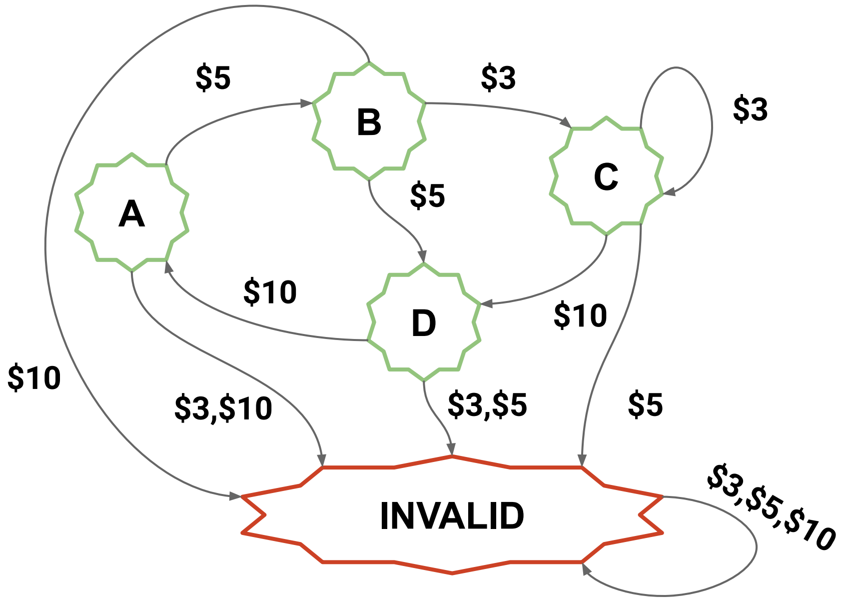
$Q_1 = \{ A, B, C, D, \text{INVALID} \}$

$\Sigma_1 = \{ \$3, \$5, \$10 \}$

$S_1 = A$

$F_1 =$

$\delta_1 =$



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

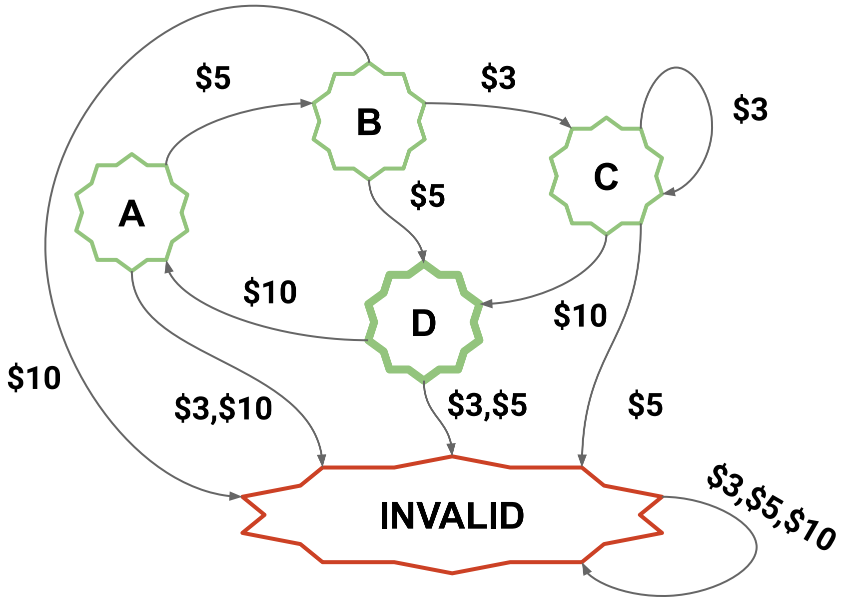
$Q_1 = \{ A, B, C, D, \text{INVALID} \}$

$\Sigma_1 = \{ \$3, \$5, \$10 \}$

$S_1 = A$

$F_1 = \{ D \}$

$\delta_1 =$



Finite Automata

Let this automata be $\boldsymbol{M_1} = (\boldsymbol{Q_1}, \boldsymbol{\Sigma_1}, \boldsymbol{\delta_1}, \boldsymbol{S_1}, \boldsymbol{F_1})$

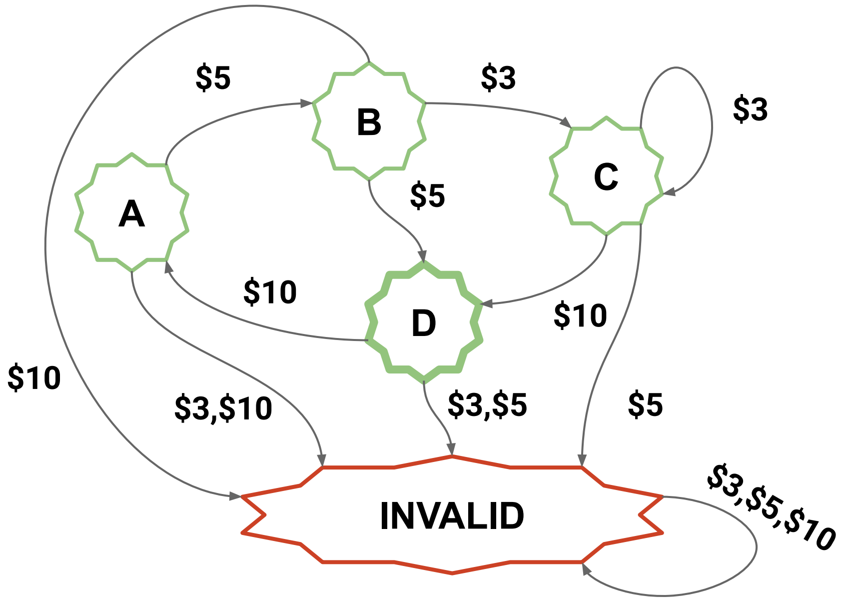
$\boldsymbol{Q_1} = \{ \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \textbf{INVALID} \}$

$\boldsymbol{\Sigma_1} = \{ \$3, \$5, \$10 \}$

$\boldsymbol{S_1} = \boldsymbol{A}$

$\boldsymbol{F_1} = \{ \boldsymbol{D} \}$

$\boldsymbol{\delta_1} = \{ ((\boldsymbol{A}, \$3), \textbf{INVALID}), ((\boldsymbol{A}, \$5), \boldsymbol{B}), \ldots \}$



Finite Automata

Let this automata be $\boldsymbol{M_1} = (\boldsymbol{Q_1}, \boldsymbol{\Sigma_1}, \boldsymbol{\delta_1}, \boldsymbol{S_1}, \boldsymbol{F_1})$

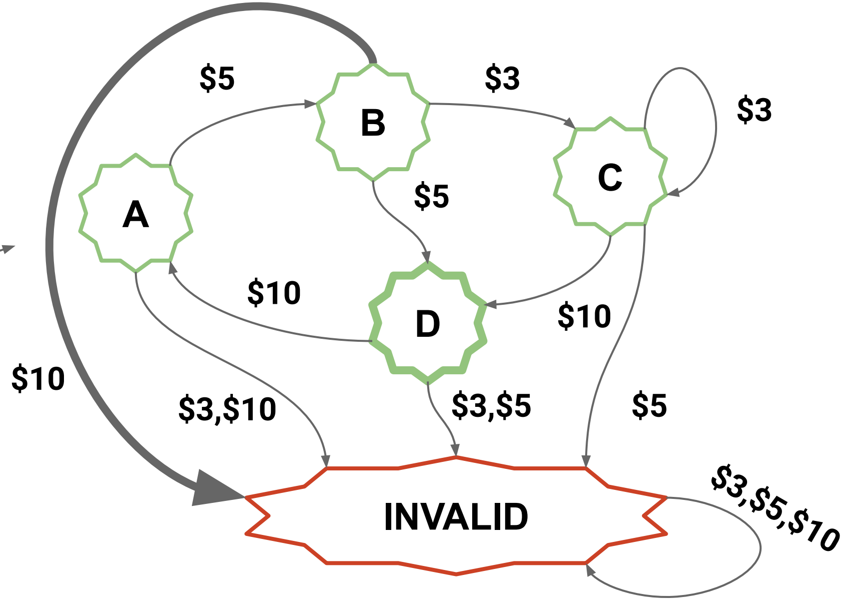
$\boldsymbol{Q_1} = \{ A, B, C, D, \text{INVALID} \}$

$\boldsymbol{\Sigma_1} = \{ \$3, \$5, \$10 \}$

$\boldsymbol{S_1} = A$

$\boldsymbol{F_1} = \{ D \}$

$\boldsymbol{\delta_1} = \{ ((A, \$3), \text{INVALID}), ((A, \$5), B), \ldots \}$



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

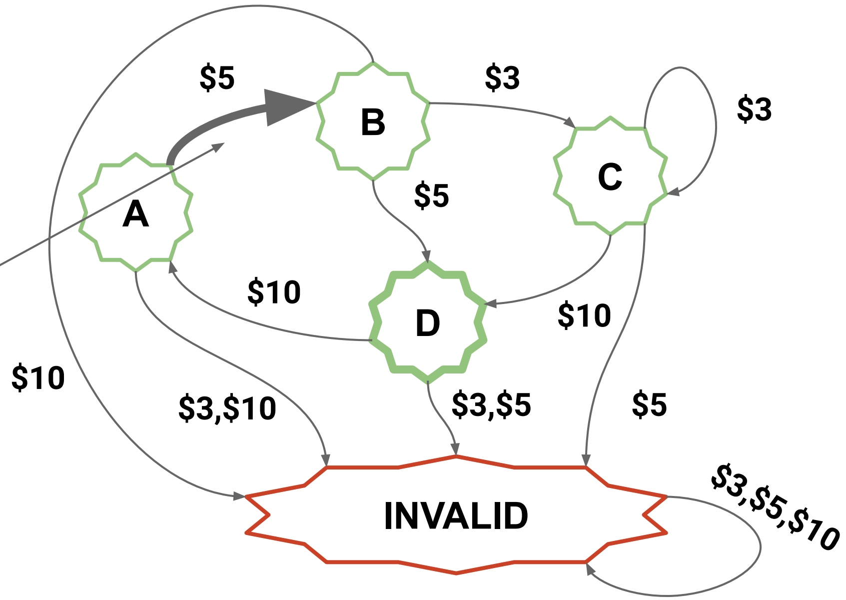
$Q_1 = \{ A, B, C, D, \text{INVALID} \}$

$\Sigma_1 = \{ \$3, \$5, \$10 \}$

$S_1 = A$

$F_1 = \{ D \}$

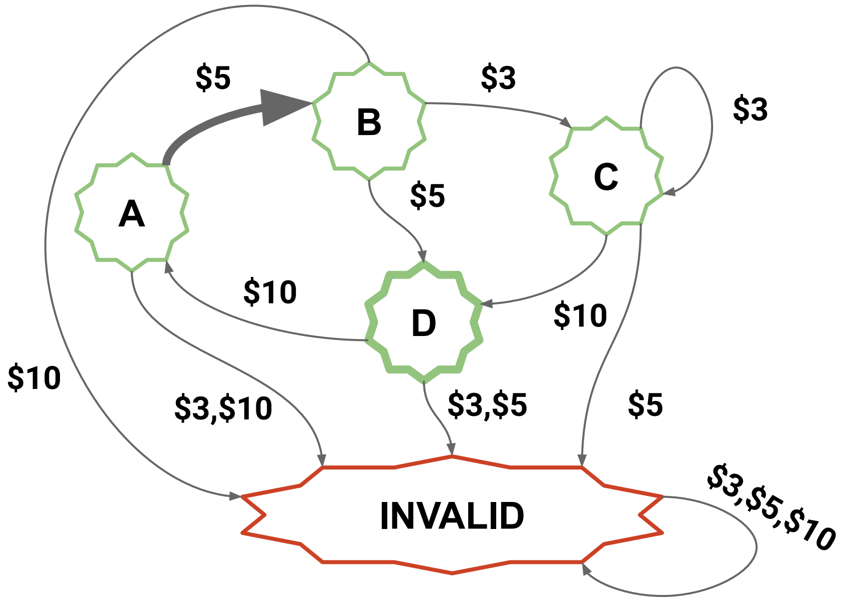
$\delta_1 = \{ ((A, \$3), \text{INVALID}), ((A, \$5), B), \ldots \}$



Finite Automata

Let this automata be $M_1 = (Q_1, \Sigma_1, \delta_1, S_1, F_1)$

		Input Symbols		
	δ_1	$3	$5	$10
States	A	INVALID	B	INVALID
	B	C	D	INVALID
	C	C	INVALID	D
	D	INVALID	INVALID	A
	INVALID	INVALID	INVALID	INVALID



Finite Automata Example

Let $\boldsymbol{M}_2 = (\boldsymbol{Q}_2, \boldsymbol{\Sigma}_2, \boldsymbol{\delta}_2, \boldsymbol{s}_0, \boldsymbol{F}_2)$ where

$\boldsymbol{Q}_2 = \{ \boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3 \}$, $\boldsymbol{\Sigma}_2 = \{ \mathbf{0}, \mathbf{1} \}$, $\boldsymbol{F} = \{ \boldsymbol{s}_0, \boldsymbol{s}_3 \}$, and $\boldsymbol{\delta}_2$ is defined as:

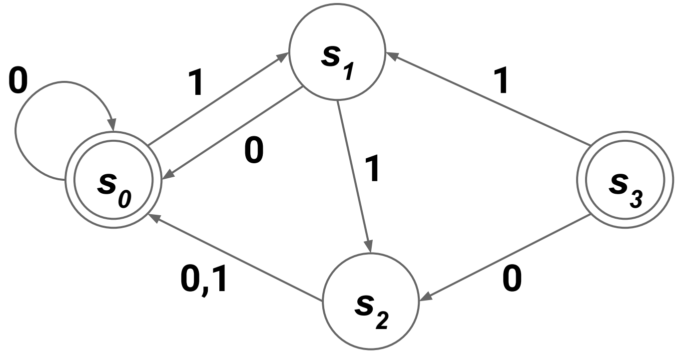
		Input Symbols	
	δ_2	**0**	**1**
States	s_0	s_0	s_1
	s_1	s_0	s_2
	s_2	s_0	s_0
	s_3	s_2	s_1

Finite Automata Example

Let $M_2 = (Q_2, \Sigma_2, \delta_2, s_0, F_2)$ where

$Q_2 = \{ s_0, s_1, s_2, s_3 \}$, $\Sigma_2 = \{ 0, 1 \}$, $F = \{ s_0, s_3 \}$, and δ_2 is defined as:

	δ_2	**Input Symbols**	
		0	**1**
States	s_0	s_0	s_1
	s_1	s_0	s_2
	s_2	s_0	s_0
	s_3	s_2	s_1



Finite Automata

A string x is **recognized** or **accepted** by the machine M if it takes the starting state q_0 to a final state.

The **language** that is ***recognized*** or ***accepted*** by the machine M, denoted by $L(M)$ is the set of strings recognized by M.

$$L(M) = \{ x \in \Sigma^* \mid \delta(q_0, x) \in F \}$$

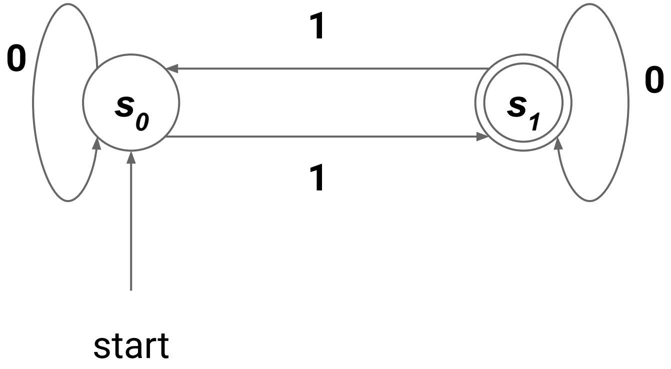
Finite Automata Examples

Problem: Design a machine that determines if the total number of 1s in a bit string is even or odd

Finite Automata Examples

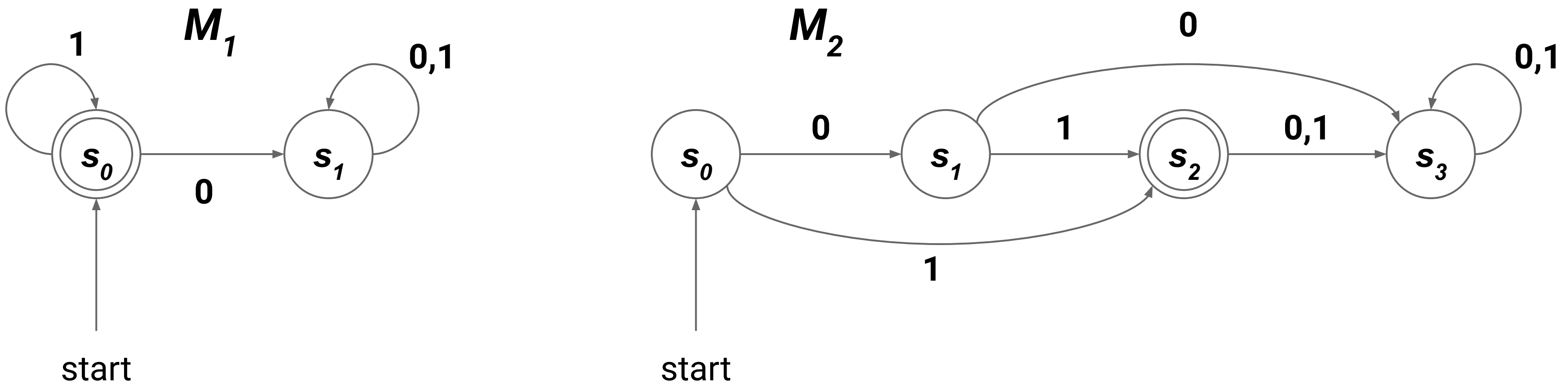
Problem: Design a machine that determines if the total number of 1s in a bit string is even or odd

If this machine accepts the string, it has odd parity. Otherwise, even.



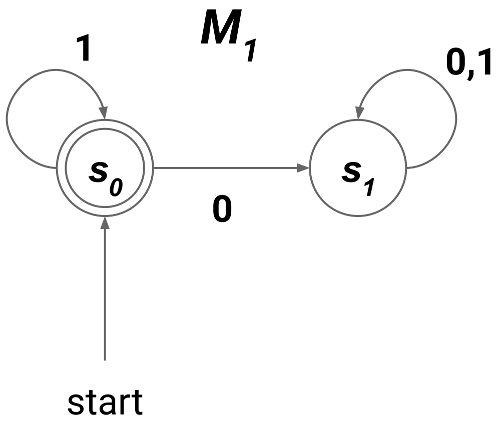
Finite Automata Examples

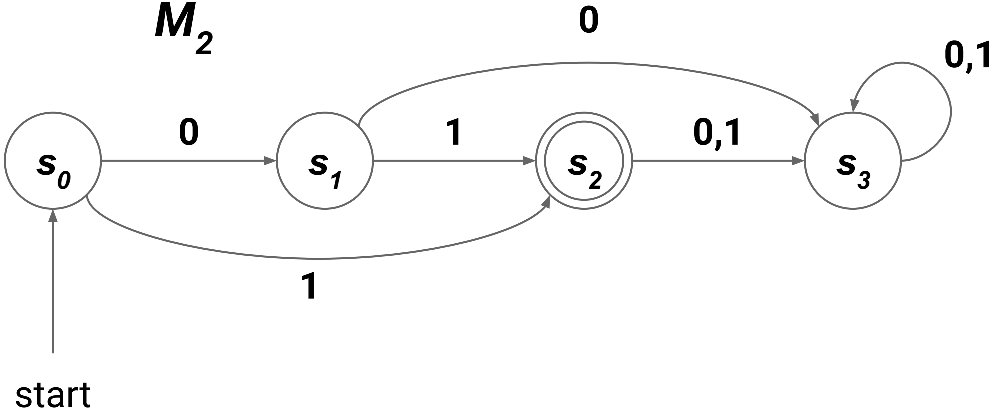
Determine the languages recognized by these finite automata



Finite Automata Examples

Determine the languages recognized by these finite automata





$L(M_1) = \{ 1, 11, 111, \ldots \} = 1^n$

$L(M_2) = \{ 01, 1 \}$

Finite Automata Examples

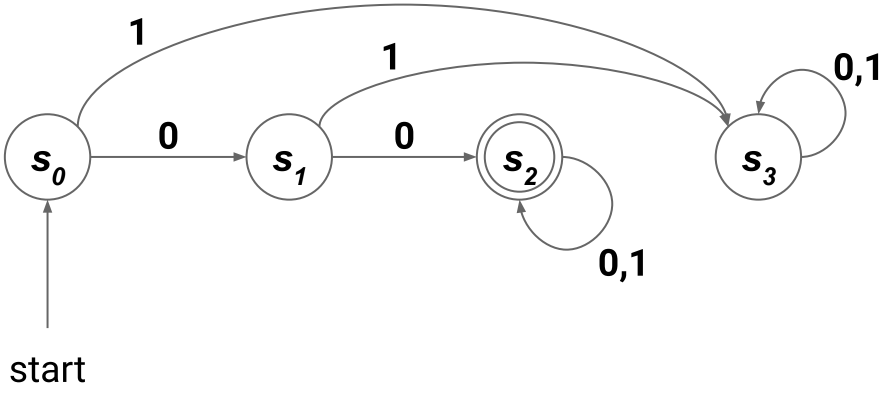
Construct finite automata that recognize the following languages:

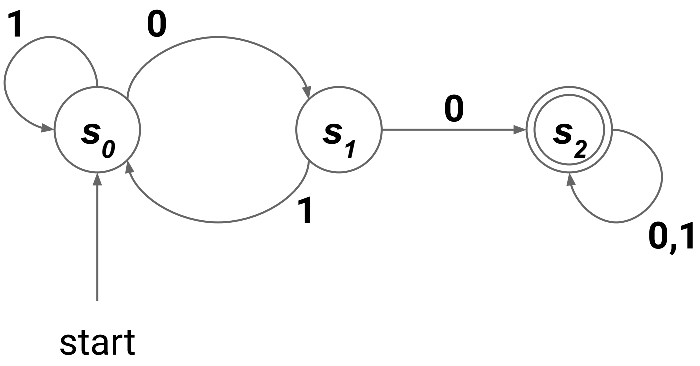
1. The set of bit strings that begin with two 0s
2. The set of bit strings that contain two consecutive 0s

Finite Automata Examples

Construct finite automata that recognize the following languages:

1. The set of bit strings that begin with two 0s
2. The set of bit strings that contain two consecutive 0s



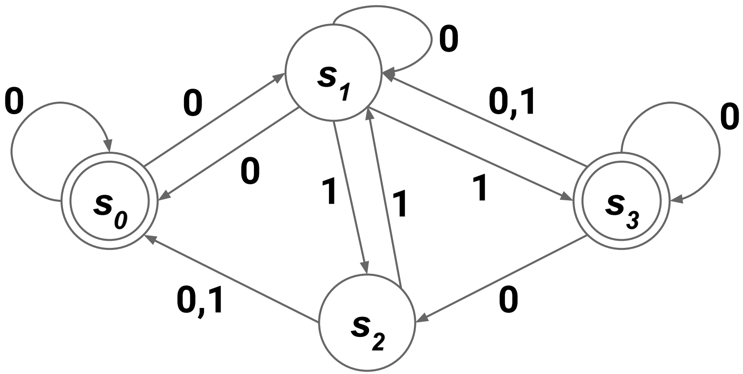


Nondeterministic Finite Automata (NFA)

A **<u>nondeterministic finite-state automaton</u>** $\boldsymbol{M} = (\boldsymbol{S}, \boldsymbol{\Sigma}, \boldsymbol{f}, \boldsymbol{s_0}, \boldsymbol{F})$ consists of a set of states $\boldsymbol{S}$, an input alphabet $\boldsymbol{\Sigma}$, and a transition function $\boldsymbol{f}$ that assigns a ***set*** of states to each pair of state and input

- DFA: For each pair of state and input there is a unique next state
- NFA: There may be many possible next states for each state/input pair

		Input Symbols	
	f	**0**	**1**
States	s_0	s_0, s_1	
	s_1	s_0, s_1	s_2, s_3
	s_2	s_0	s_0, s_1
	s_3	s_1, s_2, s_3	s_1

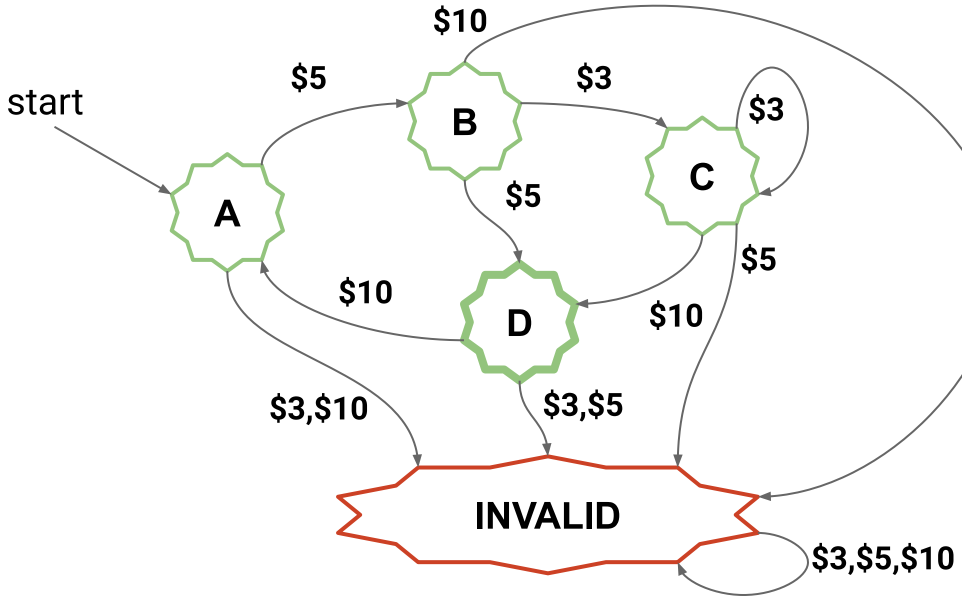


Outline

- **Computing with Limited Resources**
    - Finite Automata (Model of Computation)
    - **Regular Expressions**

Valid Fare Pattern

Is there a pattern representing every fare sequence to get from ***A*** to ***D***?

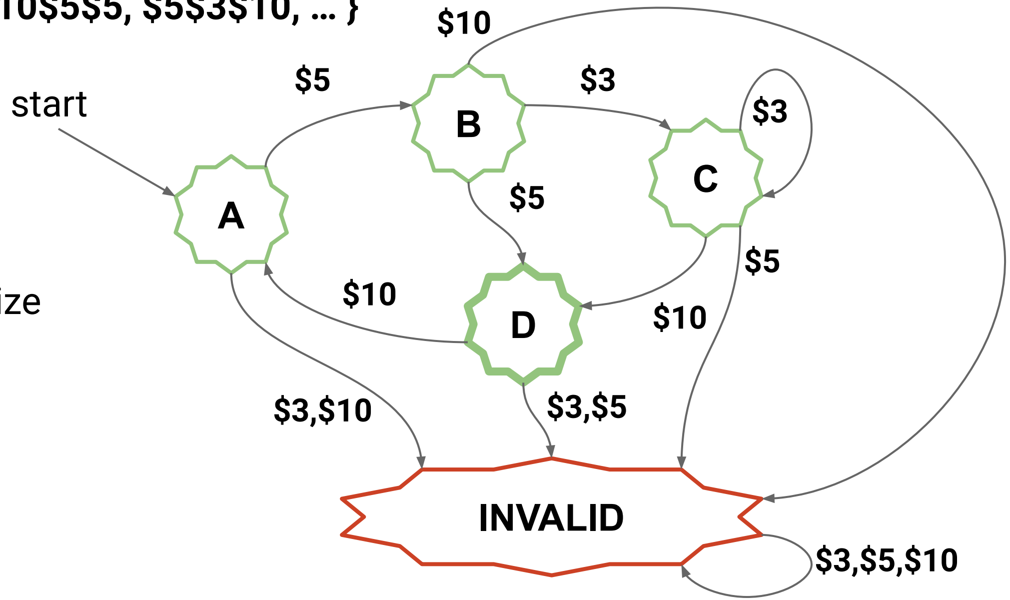


Valid Fare Pattern

Is there a pattern representing every fare sequence to get from ***A*** to ***D***?

L(***M***) **= { $5$5, $5$5$10$5$5, $5$3$10, … }**

How can we summarize the language with a succinct pattern?



Regular Expressions

A **<u>regular expression</u>**, or ***regex***, ***r*** over alphabet $\Sigma = \{ \mathbf{c_1}, \mathbf{c_2}, ..., \mathbf{c_k} \}$ is:

- $\boldsymbol{r} = \mathbf{c_i}$ for some $\boldsymbol{i} \in \{ \mathbf{1, ..., k} \}$
- $\boldsymbol{r} = \varnothing$
- $\boldsymbol{r} = \boldsymbol{\lambda}$

or, given regular expressions $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$, we can build up a new regex ***r***:

- $\boldsymbol{r} = (\boldsymbol{r_1} \mid \boldsymbol{r_2})$ ← $\boldsymbol{r_1}$ **OR** $\boldsymbol{r_2}$, also sometimes written $(\boldsymbol{r_1} \cup \boldsymbol{r_2})$
- $\boldsymbol{r} = (\boldsymbol{r_1}\boldsymbol{r_2})$ ← $\boldsymbol{r_1}$ ***concatenated with*** $\boldsymbol{r_2}$
- $\boldsymbol{r} = (\boldsymbol{r_1})^*$ ← kleene closure (0 or more repetitions)

Regular Expressions

Each regular expression represents a set specified by these rules:

- $\varnothing$ represents the empty set (the set with no strings)
- **λ** represents the set { **λ** } (the set containing the empty string)
- ***x*** represents the set { ***x*** } (the set containing one symbol, ***x***)
- (***AB***) represents the concatenation of the sets represented by ***A*** and ***B***
- (***A*** ∪ ***B***) represents the union of the sets represented by ***A*** and ***B***
- ***A****** represents the Kleene closure of the set represented by ***A***

Regular Sets are the sets represented by regular expressions

Examples

Let $\Sigma = \{ a, b \}$

Examples

Let $\Sigma = \{ a, b \}$

Let $\boldsymbol{r_1} = a$, $\boldsymbol{r_2} = b$

- $\boldsymbol{L(r_1)} = \{ a \}$, $\boldsymbol{L(r_2)} = \{ b \}$

Examples

Let $\Sigma = \{ a, b \}$

Let $\boldsymbol{r_1} = a$, $\boldsymbol{r_2} = b$

- $\boldsymbol{L(r_1)} = \{ a \}$, $\boldsymbol{L(r_2)} = \{ b \}$

Let $\boldsymbol{r_3} = (\boldsymbol{r_1} \mid \boldsymbol{r_2}) = (a \mid b)$

- $\boldsymbol{L(r_3)} = \{ a, b \} \leftarrow$ a or b

Examples

Let $\Sigma = \{ a, b \}$

Let $\boldsymbol{r_1} = a$, $\boldsymbol{r_2} = b$

- $\boldsymbol{L(r_1)} = \{ a \}$, $\boldsymbol{L(r_2)} = \{ b \}$

Let $\boldsymbol{r_3} = (\boldsymbol{r_1} \mid \boldsymbol{r_2}) = (a \mid b)$

- $\boldsymbol{L(r_3)} = \{ a, b \}$

Let $\boldsymbol{r_4} = (\boldsymbol{r_1 r_2}) = (ab)$

- $\boldsymbol{L(r_4)} = \{ ab \} \leftarrow$ a followed by b

Examples

Let Σ = { a, b }

Let $\boldsymbol{r_1}$ **=** a**,** $\boldsymbol{r_2}$ = b

- $\boldsymbol{L(r_1)}$ **=** { a }**,** $\boldsymbol{L(r_2)}$ **=** { b }

Let $\boldsymbol{r_3}$ = ($\boldsymbol{r_1}$ | $\boldsymbol{r_2}$) = (a | b)

- $\boldsymbol{L(r_3)}$ **=** { a, b }

Let $\boldsymbol{r_4}$ = ($\boldsymbol{r_1 r_2}$) = (ab)

- $\boldsymbol{L(r_4)}$ **=** { ab }

Let $\boldsymbol{r_5}$ **= (**$\boldsymbol{r_1}$**)*** = (a)*

- $\boldsymbol{L(r_5)}$ **=** { **λ,** a, aa, aaa, … }

0 or more copies of a concatenated

Examples

Let $\Sigma = \{ a, b \}$

Let $\boldsymbol{r_1} = a$, $\boldsymbol{r_2} = b$
- $L(r_1) = \{ a \}$, $L(r_2) = \{ b \}$

Let $\boldsymbol{r_3} = (r_1 \mid r_2) = (a \mid b)$
- $L(r_3) = \{ a, b \}$

Let $\boldsymbol{r_4} = (r_1 r_2) = (ab)$
- $L(r_4) = \{ ab \}$

Let $\boldsymbol{r_5} = (r_1)^* = (a)^*$
- $L(r_5) = \{ \boldsymbol{\lambda}, a, aa, aaa, \ldots \}$

Let $\boldsymbol{r_6} = \varnothing$
- $L(r_6) = \{ \}$ ← matches nothing

Examples

Let Σ = { a, b }

Let r_1 = a, r_2 = b

- $L(r_1)$ = { a }, $L(r_2)$ = { b }

Let r_3 = (r_1 | r_2) = (a | b)

- $L(r_3)$ = { a, b }

Let r_4 = ($r_1 r_2$) = (ab)

- $L(r_4)$ = { ab }

Let r_5 = (r_1)* = (a)*

- $L(r_5)$ = { **λ,** a, aa, aaa, … }

Let r_6 = ∅

- $L(r_6)$ = { }

Let r_7 = **λ**

- $L(r_7)$ = **{ λ }** ←matches **λ**