

Regular Expression (RE)

- Equivalent to regular grammars and finite automata.
- used to implement scanners.

RE Definition

- If a and b are regular expressions:
 - Concatenation – $a \cdot b$
is a regular expression consisting of RE a followed by RE b .
 - Alternation – $a \mid b$
is a regular expression consisting of either RE a or RE b .
 - Kleene Closure – a^*
is a regular expression consisting of 0 or more instances of RE a concatenated together.

The Empty RE – ϵ

- The RE with no symbols in it is ϵ .
- For example, 0 instances of a^* is ϵ .

RE Examples

If l is a letter and d is a digit,

- $l \cdot (l \mid d)^*$ – is an RE for an identifier. (l , ld , ll , ldd , ldl , lld , lll , $lddd$, $lddl$, $ldld$, $ldll$, ...).
- d^* – is a bad RE for an integer (ϵ , d , dd , ddd , ...).
- $d \cdot d^*$ – is a good RE for an integer (d , dd , ddd , ...).
$$d \cdot d^* \equiv dd^* \equiv d^+$$
- $d^* . d^*$ (d^* followed by a period followed by d^*) – is a bad RE for a real number ($., .d$, $d.$, $d.d$, $dd.d$, $d.dd$, $dd.$, $.dd$, $dd.dd$, ...).

Signed/Unsigned Integer/Real Number

$(+|-|\epsilon)(d^+|d^+.d^*|d^*.d^+)$

Notice that parentheses are frequently included (but are ignored) and the concatenation dot is frequently omitted.

If $d = (0|1|2|3|4|5|6|7|8|9)$

The RE recognizes (23, -4, .137, 11803., +843.31, 59203.11765,...)

Identifier with Underscores

$$l (l | d)^* (_ (l | d)^+)^*$$

Where $l = (a | b | c | d | \dots | x | y | z)$

$$d = (0 | 1 | 2 | \dots | 8 | 9)$$

The RE recognizes (u, p1, apple, var_1,
first_var_24, ...)

Finite State Automata (FSA)

- FSA = four tuple = $(S, \Sigma, S_0, F, \delta)$ where:
 - S = set of states = {states}
 - Σ = set of input tokens = {tokens}
 - S_0 = start state
 - F = set of final states {final states}
 - δ = mapping function

FSA Example – Identifiers

$$S = \{ \textcircled{A}, \textcircled{B} \}$$

$$S_0 = \textcircled{A}$$

$$F = \{ \textcircled{B} \}$$

$$\Sigma = \{l, d\} \quad \text{Where}$$

$$l = \{a \mid b \dots \mid z\}$$

$$d = \{0 \mid 1 \dots 9\}$$

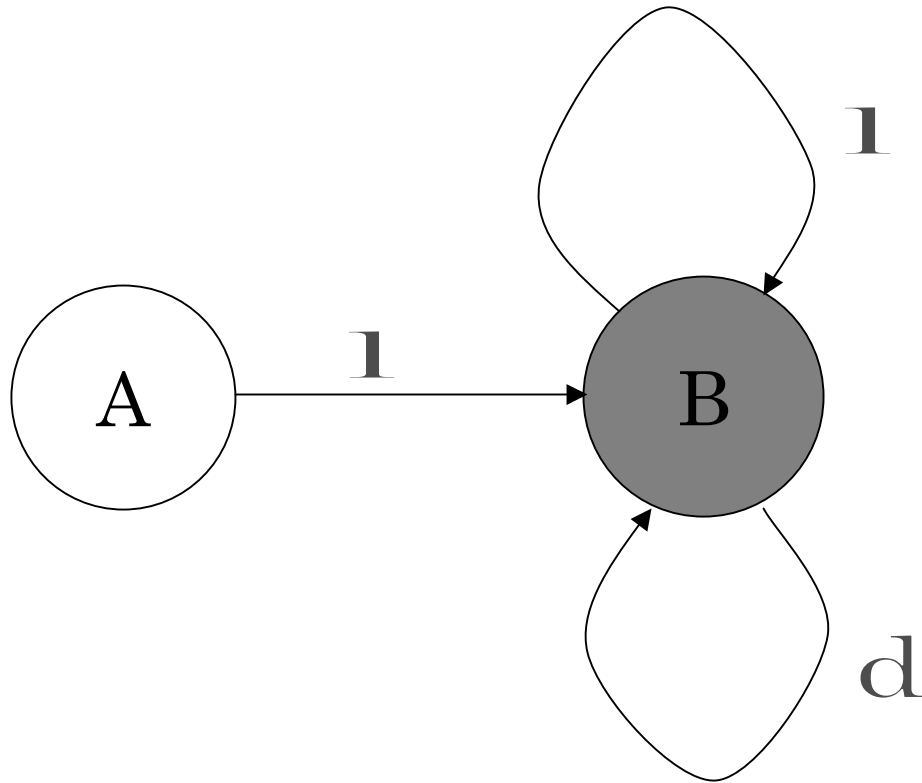
$$\delta(\textcircled{A}, l) = \textcircled{B}$$

$$\delta(\textcircled{B}, l) = \textcircled{B}$$

$$\delta(\textcircled{A}, d) = \Phi$$

$$\delta(\textcircled{B}, d) = \textcircled{B}$$

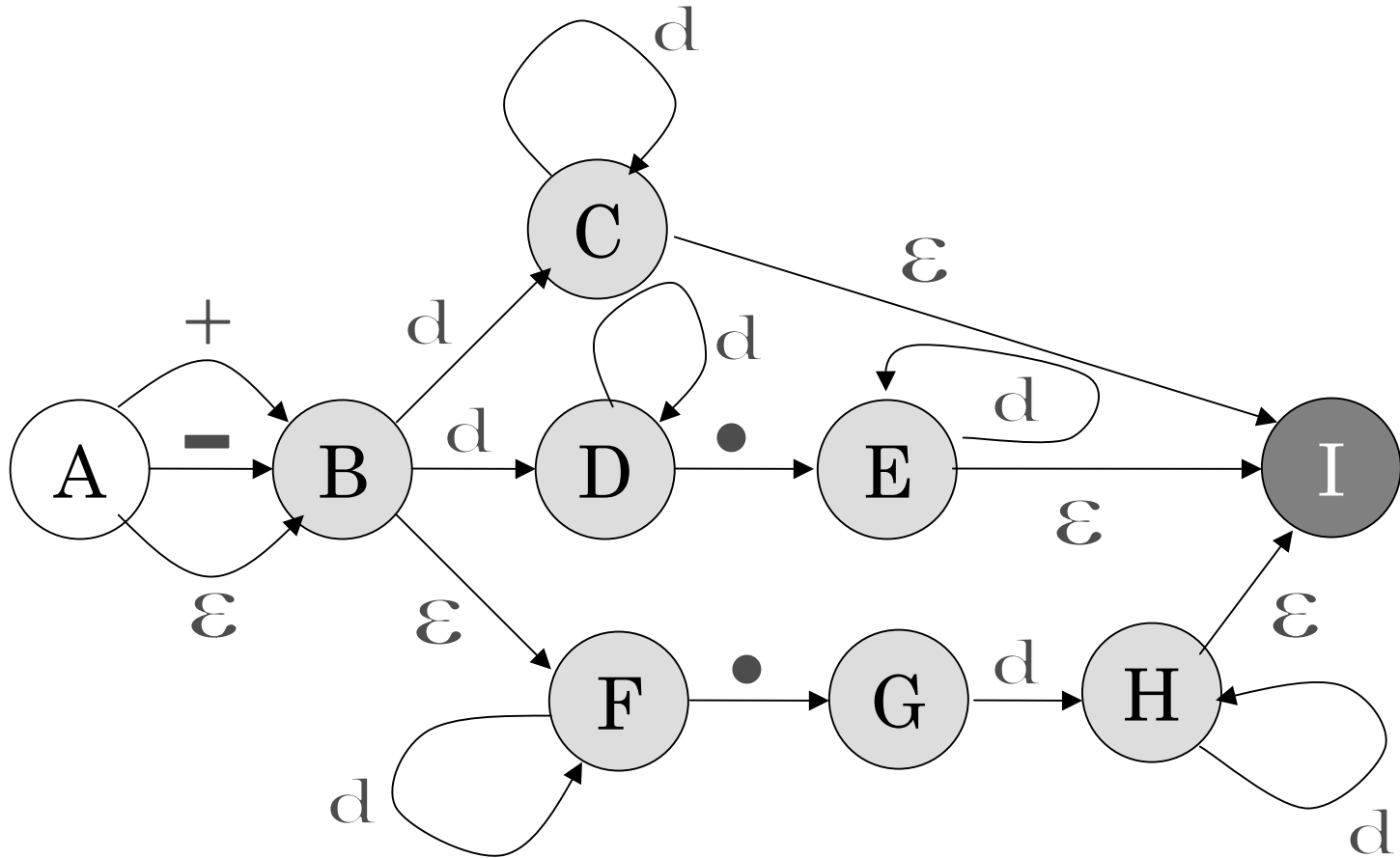
FSA State Diagram



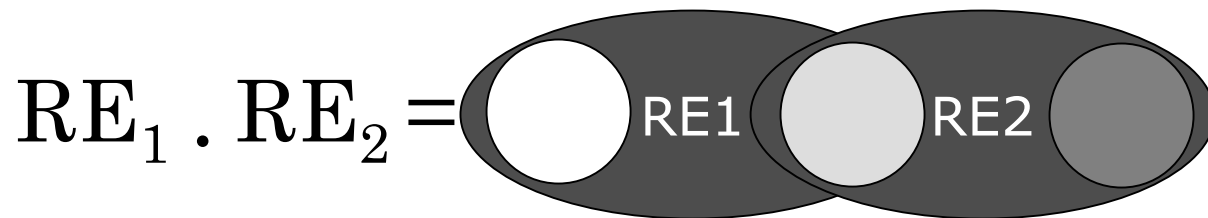
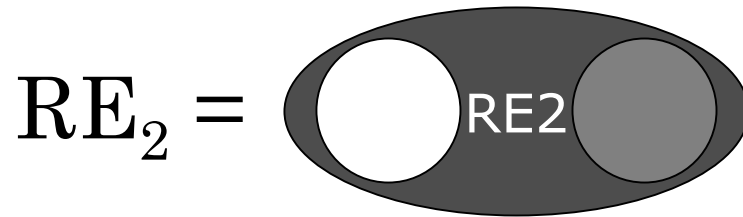
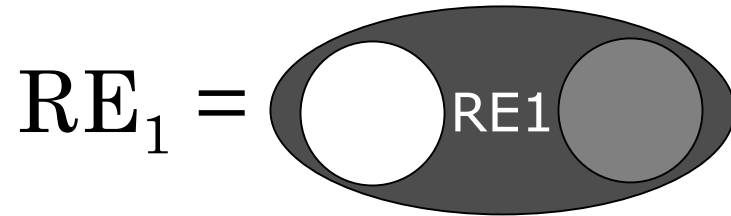
FSA State Table

	l	d
A	B	
B	B	B

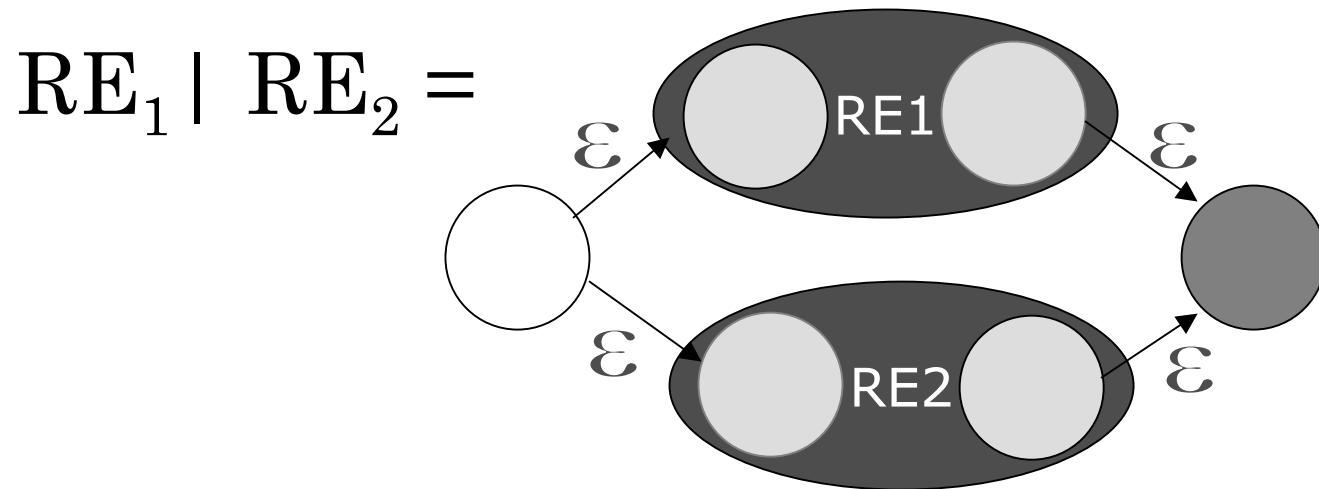
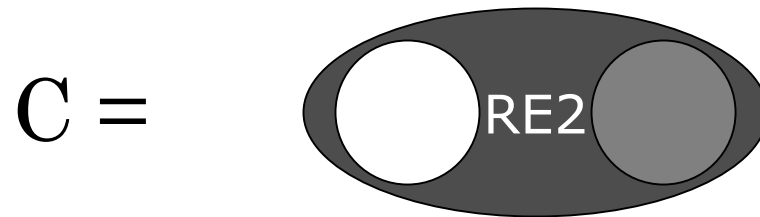
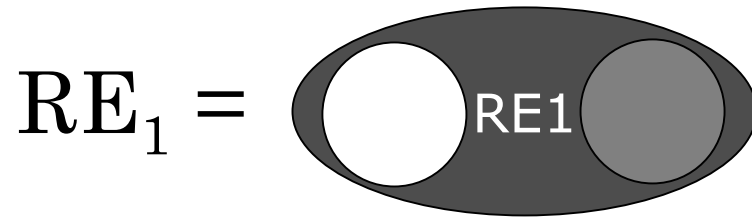
$$RE_1 = (+ | - | \varepsilon)(d^+ | d^+ \cdot d^* | d^* \cdot d^+)$$



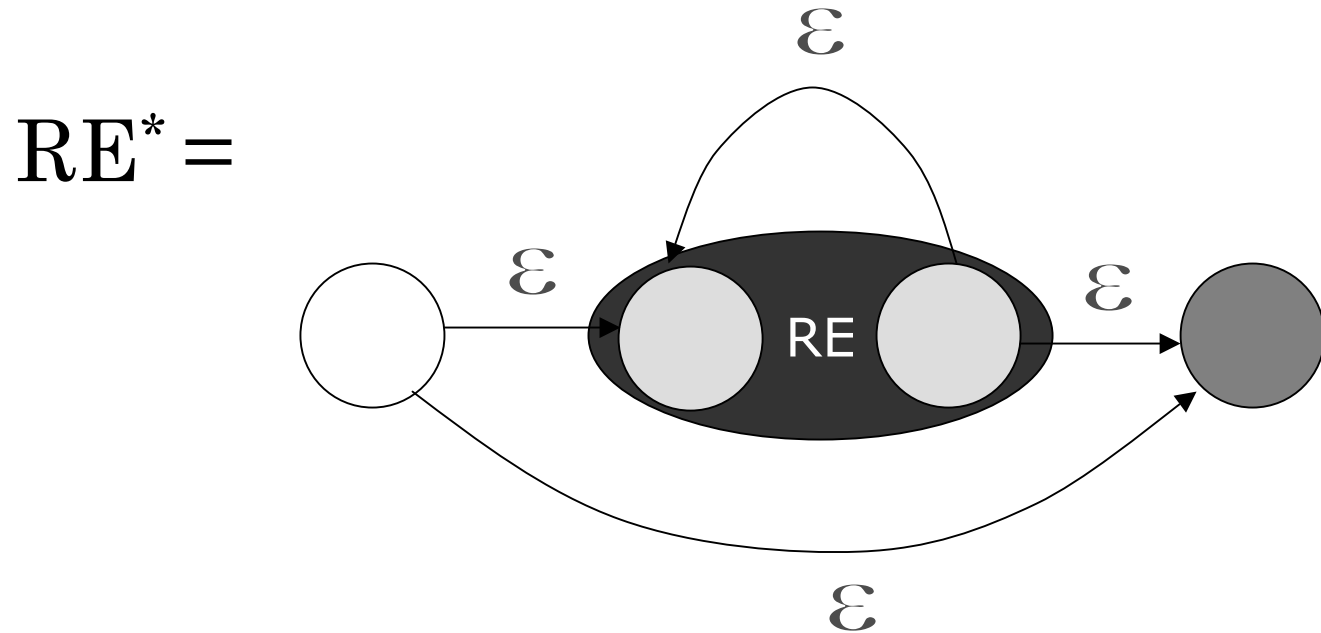
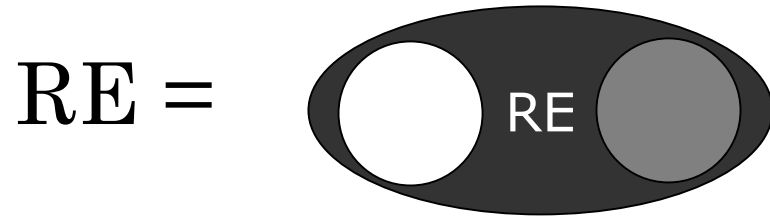
RE \rightarrow NDFSA – Concatenation



RE \rightarrow NDFSA – Alternation



RE \rightarrow NDFSA – Closure



State Table for RE_1

	d	+	-	.	ϵ
A		B	B		B
B	CD				F
C	C				I
D	D			E	
E	E				I
F	F			G	
G	H				
H	H				I
I					

ϵ -Closure

- Start with NDFSA
- Used to convert NDFSA to DFSA

ϵ -Closure Method

- New start state is the set of states containing the old start state and all states that can be reached from the old start state through only ϵ moves.
- New states in DFSA are defined by the set of states that can be reached from another set of states on a move over an input symbol followed by moves over ϵ .
- If the set of new states contains a final state, the new state is a final state; otherwise, it is a non-final state.

ε -Closure Example

Consider RE_1 :

$$\text{Start} = \{A, B, F\} = 1$$

$$\delta(\{A, B, F\}, d) = \delta(1, d) = \{C, D, F, I\} = 2$$

$$\delta(\{A, B, F\}, +) = \delta(1, +) = \{B, F\} = 3$$

$$\delta(\{A, B, F\}, -) = \delta(1, -) = \{B, F\} = 3$$

$$\delta(\{A, B, F\}, \cdot) = \delta(1, \cdot) = \{G\} = 4$$

$$\delta(\{C, D, F, I\}, d) = \delta(2, d) = (\{C, D, F, I\} = 2$$

$$\delta(\{C, D, F, I\}, +) = \delta(2, +) = \Phi$$

$$\delta(\{C, D, F, I\}, -) = \delta(2, -) = \Phi$$

$$\delta(\{C, D, F, I\}, \cdot) = \delta(2, \cdot) = \{E, G, I\} = 5$$

ε -Closure Example (Cont)

$$\delta(\{B, F\}, d) = \delta(3, d) = \{C, D, F, I\} = 2$$

$$\delta(\{B, F\}, +) = \delta(3, +) = \delta(\{B, F\}, -) = \delta(3, -) = \Phi$$

$$\delta(\{B, F\}, \cdot) = \delta(3, \cdot) = \{G\} = 4$$

$$\delta(\{G\}, d) = \delta(4, d) = \{H, I\} = 6$$

$$\delta(\{G\}, +) = \delta(4, +) = \delta(4, -) = \delta(4, \cdot) = \Phi$$

$$\delta(\{E, G, I\}, d) = \delta(5, d) = \{E, H, I\} = 7$$

$$\delta(\{E, G, I\}, +) = \delta(5, +) = \delta(5, -) = \delta(5, \cdot) = \Phi$$

$$\delta(\{H, I\}, d) = \delta(6, d) = \{H, I\} = 6$$

$$\delta(\{H, I\}, +) = \delta(6, +) = \delta(6, -) = \delta(6, \cdot) = \Phi$$

$$\delta(\{E, H, I\}, d) = \delta(7, d) = \{E, H, I\} = 7$$

$$\delta(\{E, H, I\}, +) = \delta(7, +) = \delta(7, -) = \delta(7, \cdot) = \Phi$$

RE₁ DFSA State Table

	d	+	-	.
1	2	3	3	4
2	2			5
3	2			4
4	6			
5	7			
6	6			
7	7			

Redundant States

- The DFSA generated by the ϵ -Closure method could contain redundant states.
- Determine and remove redundant states by partitioning.

Partitioning

- Divide the DFSA states into
 1. The set of non-final states
 2. The set of final states
- Two states in the same partition can be placed in different partitions if:
 1. For any input symbol, one goes to a new state and the other goes to Φ .
 2. For any input symbol, one goes to one partition and the other goes to another partition.

RE₁ Partitioning Example

Initial partition: {1, 3, 4} {2, 5, 6, 7}

Partition 1: {1} {3, 4} {2, 5, 6, 7}

because of rule 1 on + and -

Partition 2: {1} {3} {4} {2, 5, 6, 7}

because of rule 1 on .

Partition 3: {1} {3} {4} {2} {5, 6, 7}

because of rule 1 on .

States 5, 6, 7 all go to the same partition {5, 6, 7} on an input of d and all go to Φ on an input of +, -, and .

Therefore, by rule 2 Partition 3 is the final partition and states 5, 6, 7 are redundant.

State Table Update

- Arbitrarily select state 5 from among the redundant states 5, 6, 7.
- Eliminate state 6, 7 rows from RE_1 state table, and wherever occurring in remaining rows, replace 6, 7 with 5.

RE₁ Minimum State DFSA State Table

	d	+	-	.
1	2	3	3	4
2	2			5
3	2			4
4	5			
5	5			

RE₁ Minimum State DFSA State Diagram

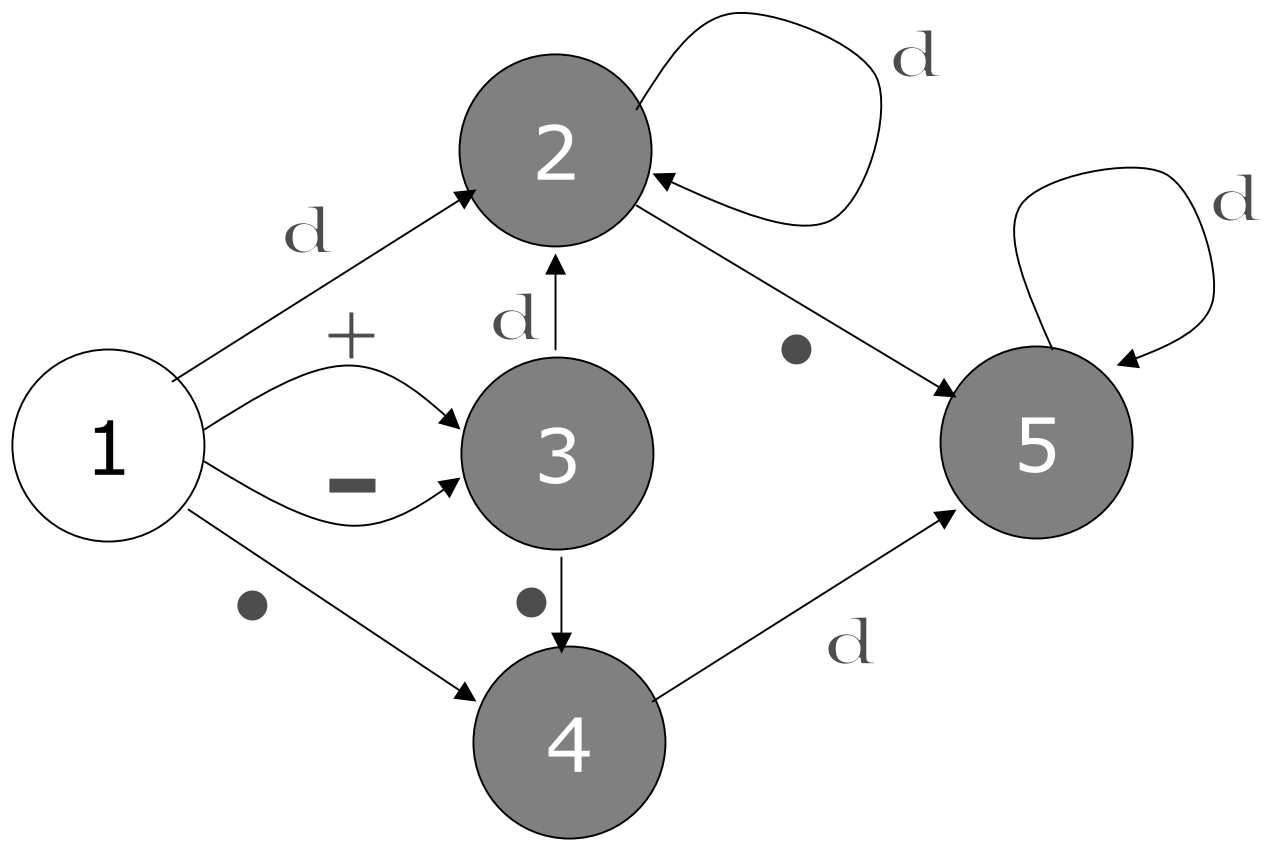


Table Method

1. Make any state that goes to a final state on ε move a final state.
2. For any move from state₁ to state₂ on ε , transfer the moves on all inputs (except ε) from the state₂ row to the state₁ row. This eliminates moves on ε .
3. Add sets of states and moves from them to state table.
4. Mark reachable states.
5. Eliminate redundant states (using partitioning).

RE₁ State Table

	d	+	-	.	ϵ
A		B	B		B
B	CD				F
C	C				I
D	D			E	
E	E				I
F	F			G	
G	H				
H	H				I
I					

Step 1

	d	+	-	.	ϵ
A		B	B		B
B	CD				F
C	C				I
D	D			E	
E	E				I
F	F			G	
G	H				
H	H				I
I					

Step 2

	d	+	-	.	ϵ
A	CDF	B	B	G	B
B	CDF			G	F
C	C				I
D	D			E	
E	E				I
F	F			G	
G	H				
H	H				I
I					

Step 3 and Step 4

	d	+	-	.
\sqrt{A}	CDF	B	B	G
\sqrt{B}	CDF			G
C	C			
D	D			E
E	E			
F	F			G
\sqrt{G}	H			
\sqrt{H}	H			
I				
$\sqrt{\text{CDF}}$	CDF			EG
\sqrt{EG}	EH			
\sqrt{EH}	EH			

DFSA State Table

	d	+	-	.
A	CDF	B	B	G
B	CDF			G
G	H			
H	H			
CDF	CDF			EG
EG	EH			
EH	EH			

Step 5

Eliminate redundant states using partitioning:

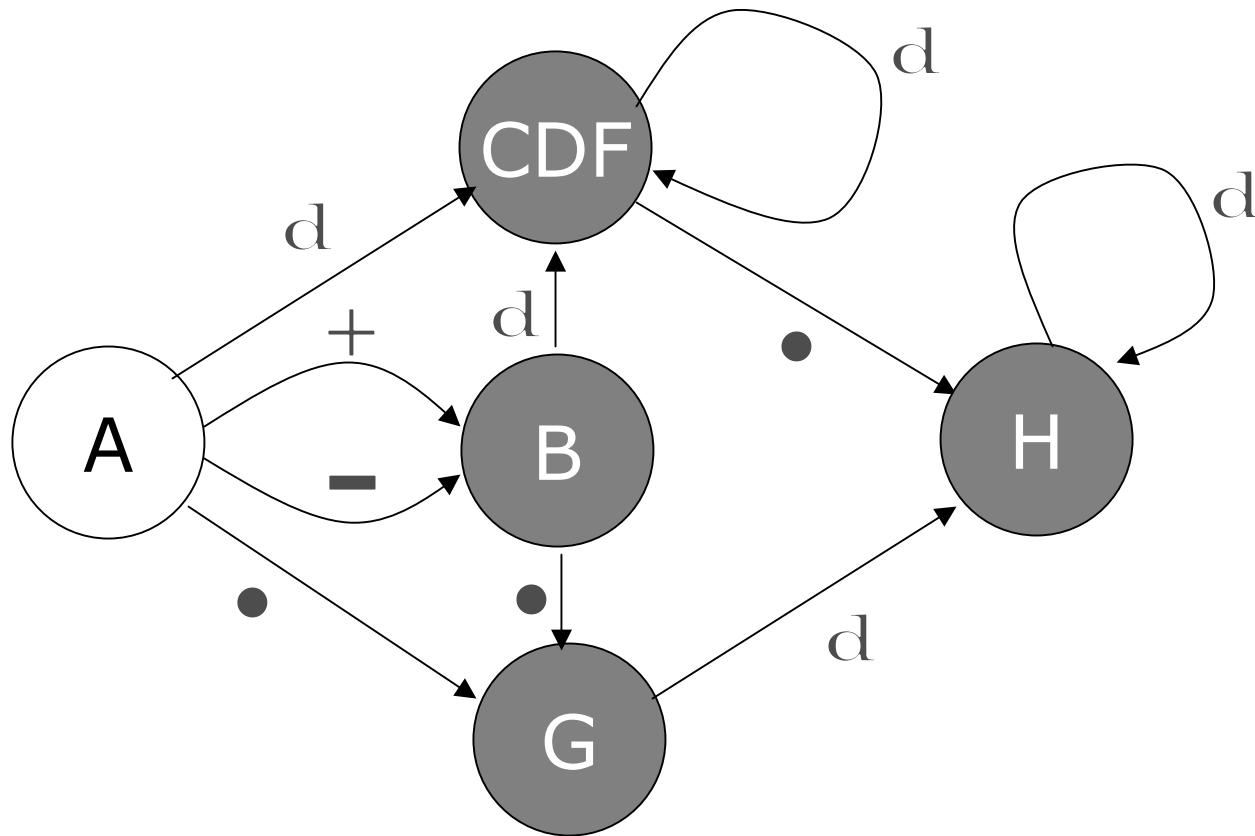
$\{A, B, G\}$ $\{H, CDF, EG, EH\}$

A is not redundant with B, G because of Rule 2 move on B. B is not redundant with G because of Rule 2 move on period. CDF is not redundant with H, EG, EH due to move on period. H, EG, EH are redundant.

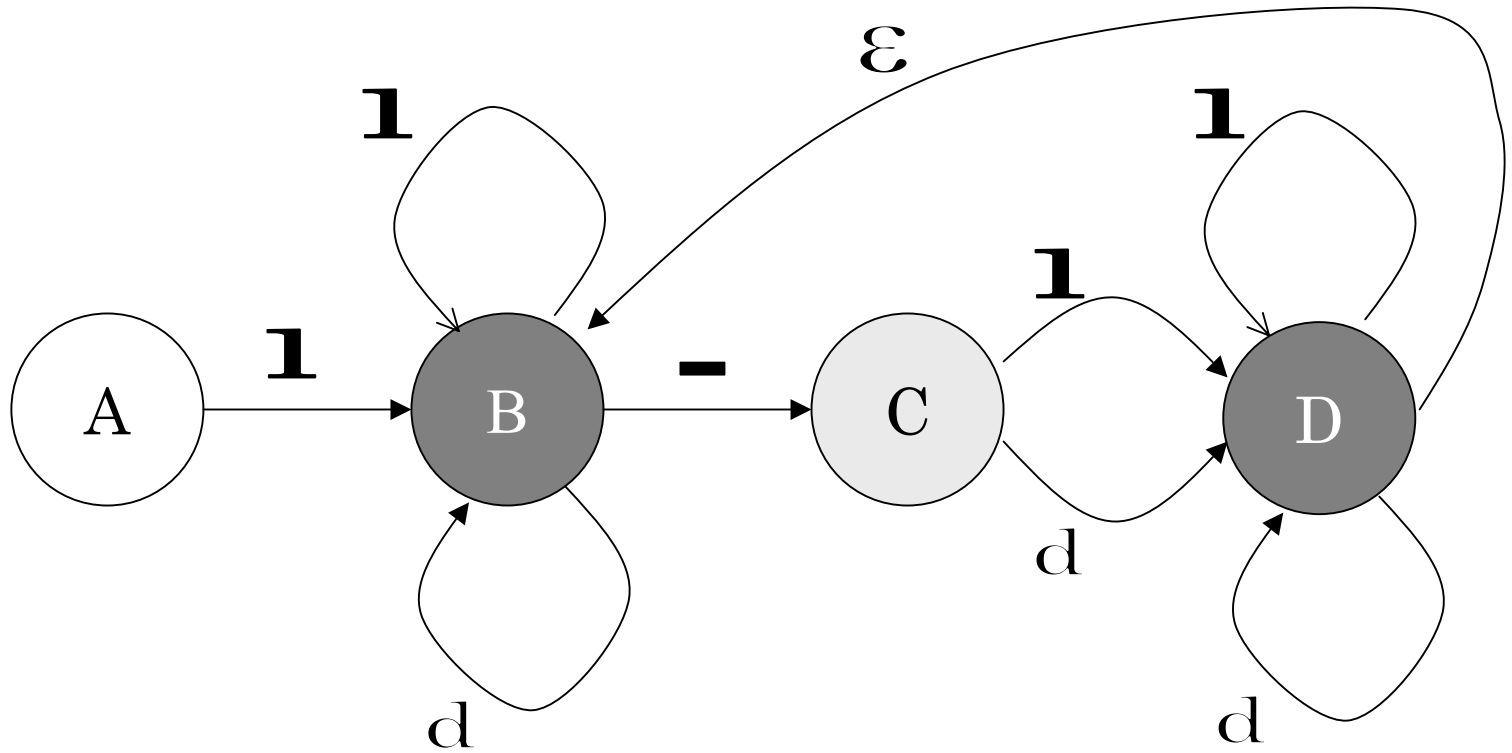
Minimum State DFSA State Table

	d	+	-	.
A	CDF	B	B	G
B	CDF			G
G	H			
H	H			
CDF	CDF			H

Minimum State DFSA State Diagram



$$RE_2 = (1 | d)^* (_ (1 | d)^+)^*$$



RE₂ ε-Closure

New start state = {A} = 1

$$\delta(1,1) = \delta(\{A\}, 1) = \{B\} = 2$$

$$\delta(2,1) = \delta(\{B\}, 1) = \{B\} = 2$$

$$\delta(2,d) = \delta(\{B\}, d) = \{B\} = 2$$

$$\delta(2,_) = \delta(\{B\}, _) = \{C\} = 3$$

$$\delta(3,1) = \delta(\{C\}, 1) = \{B,D\} = 4$$

$$\delta(3,d) = \delta(\{C\}, d) = \{B,D\} = 4$$

$$\delta(4,1) = \delta(\{B,D\}, 1) = \{B,D\} = 4$$

$$\delta(4,d) = \delta(\{B,D\}, d) = \{B,D\} = 4$$

$$\delta(4,_) = \delta(\{B,D\}, _) = \{C\} = 3$$

RE₂ DFSA State Table

	1	d	–
1	2		
2	2	2	3
3	4	4	
4	4	4	3

Partition

$\{1,3\}$ $\{2,4\}$

$\{1\}$ $\{3\}$ $\{2,4\}$ due to move from 1 and 3 on d

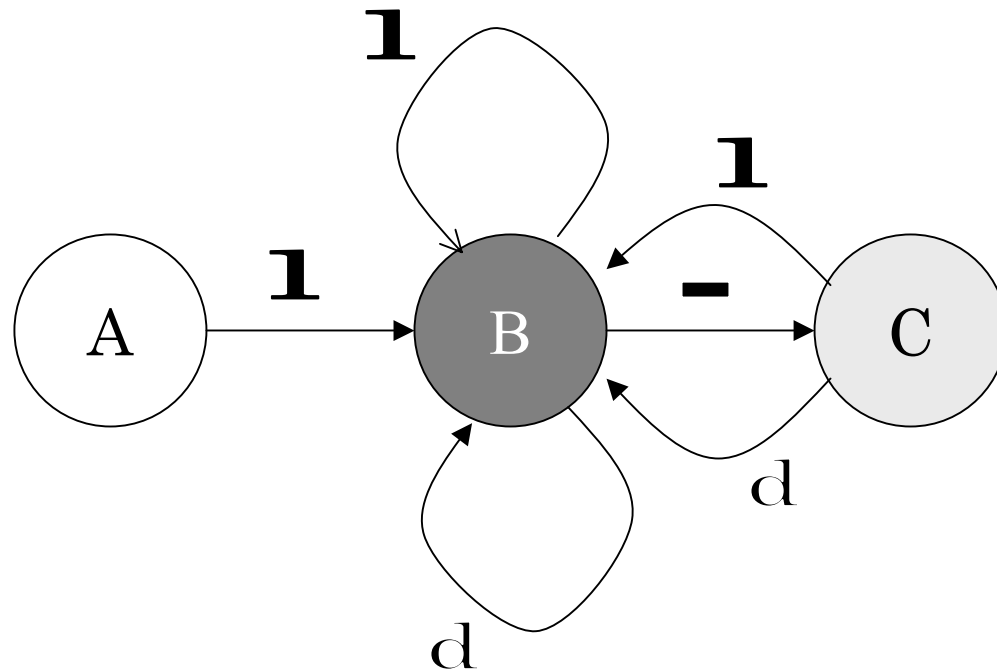
Since we cannot partition 2 and 4, they are redundant.

Remove row 4 and replace 4 with 2 wherever
Appearing in state table.

RE₂ Minimum State DFSA State Table

	1	d	–
1	2		
2	2	2	3
3	2	2	

RE₂ Minimum State DFSA State Diagram



Code

```
void b() {
char x; x = getchar();
    if(x == '\l' || x == 'd') b();
    elseif(x == '_') c();
    else return ID; }
void c() {
char x; x = getchar();
    if(x = '\l' || x = 'd') b();
    else return error; }
void main () {
    if (x = getchar() == '\l') b(); else return
error; }
```