Automata Formal Languages & Logic

Preet Kanwal

Department of Computer Science & Engineering

Automata Formal Languages & Logic

Unit 2

Preet Kanwal

Department of Computer Science & Engineering

Regex is an algebraic way to describe regular languages.

Regular Expression

Atoms
- A Character/symbol
- ϕ (L = { })
- λ (L = {λ})

Operators
- (R)
- R*
- $R_1 . R_2$ or $R_1 R_2$
- $R_1 + R_2$ or $R_1 | R_2$

Example

$$a \quad b^* \quad c + d$$

Example

$$(a \quad b* \quad c)+ d$$

R_1

R_2

or

Example

$$(a \quad b^* \quad c) + d$$

Starting with an 'a'

Followed by 0 or more no. of b's

Ending with a 'c'

Example

$$(a \quad b* \quad c)+ \, d$$

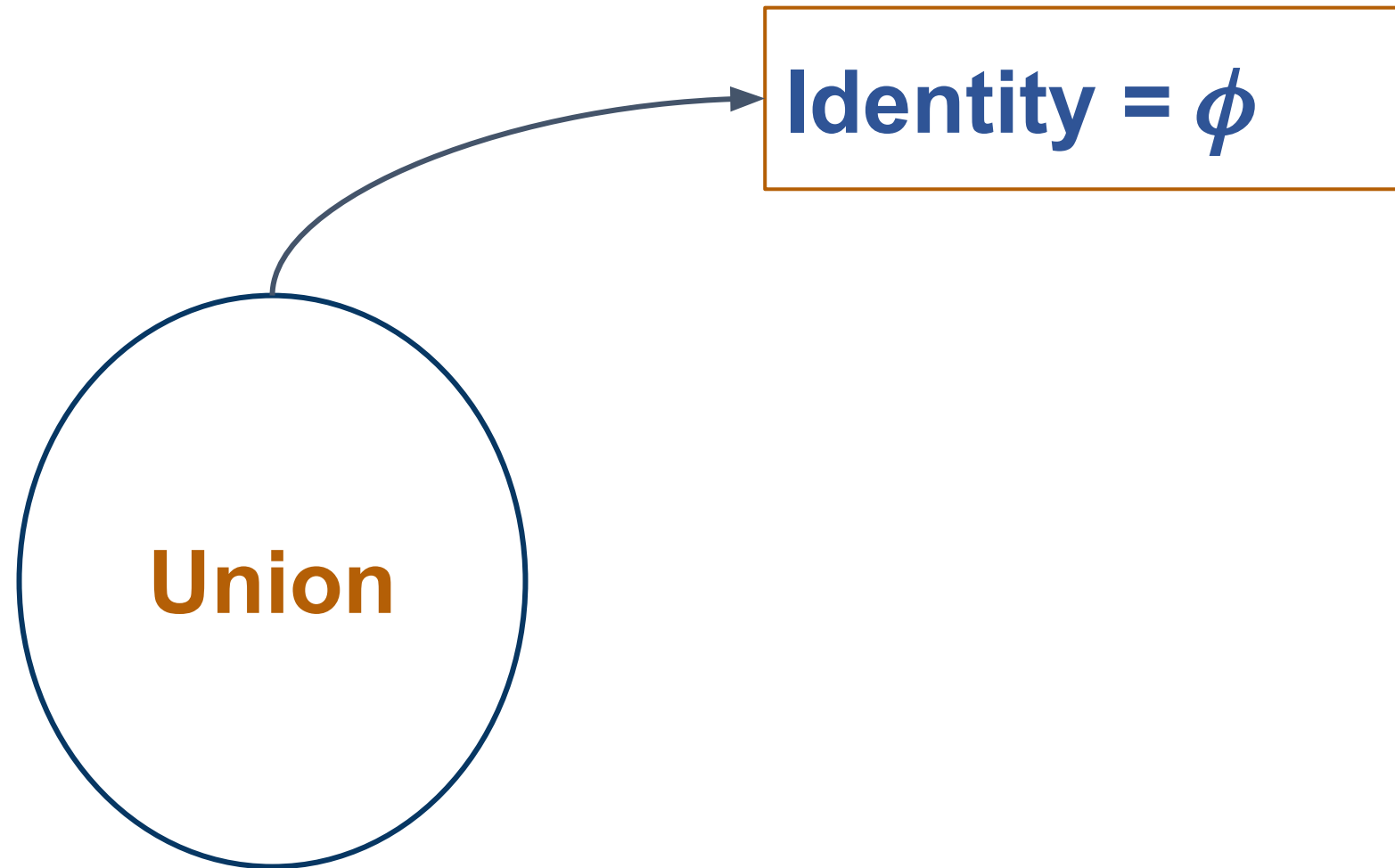String that contains only d

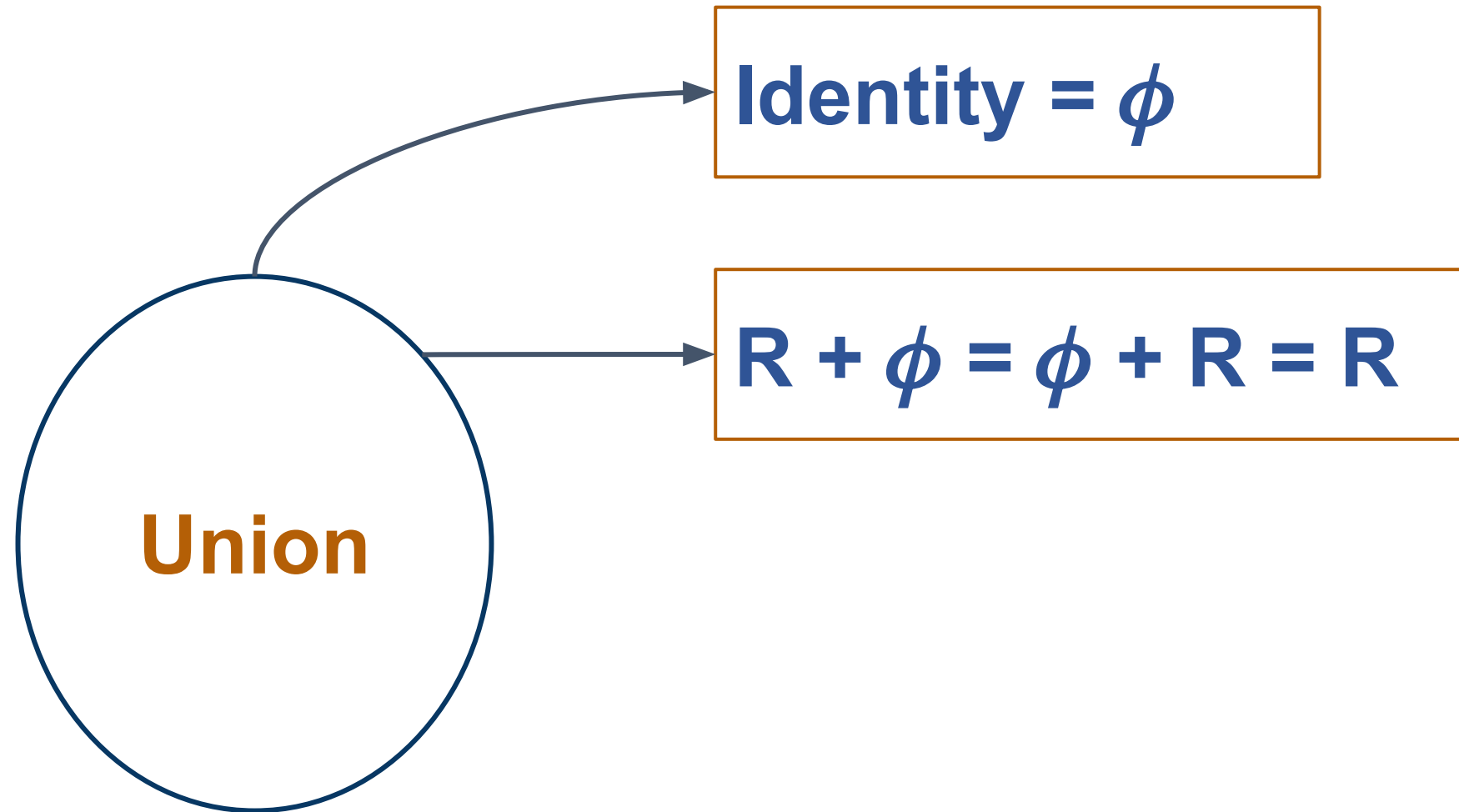Starting with an 'a'

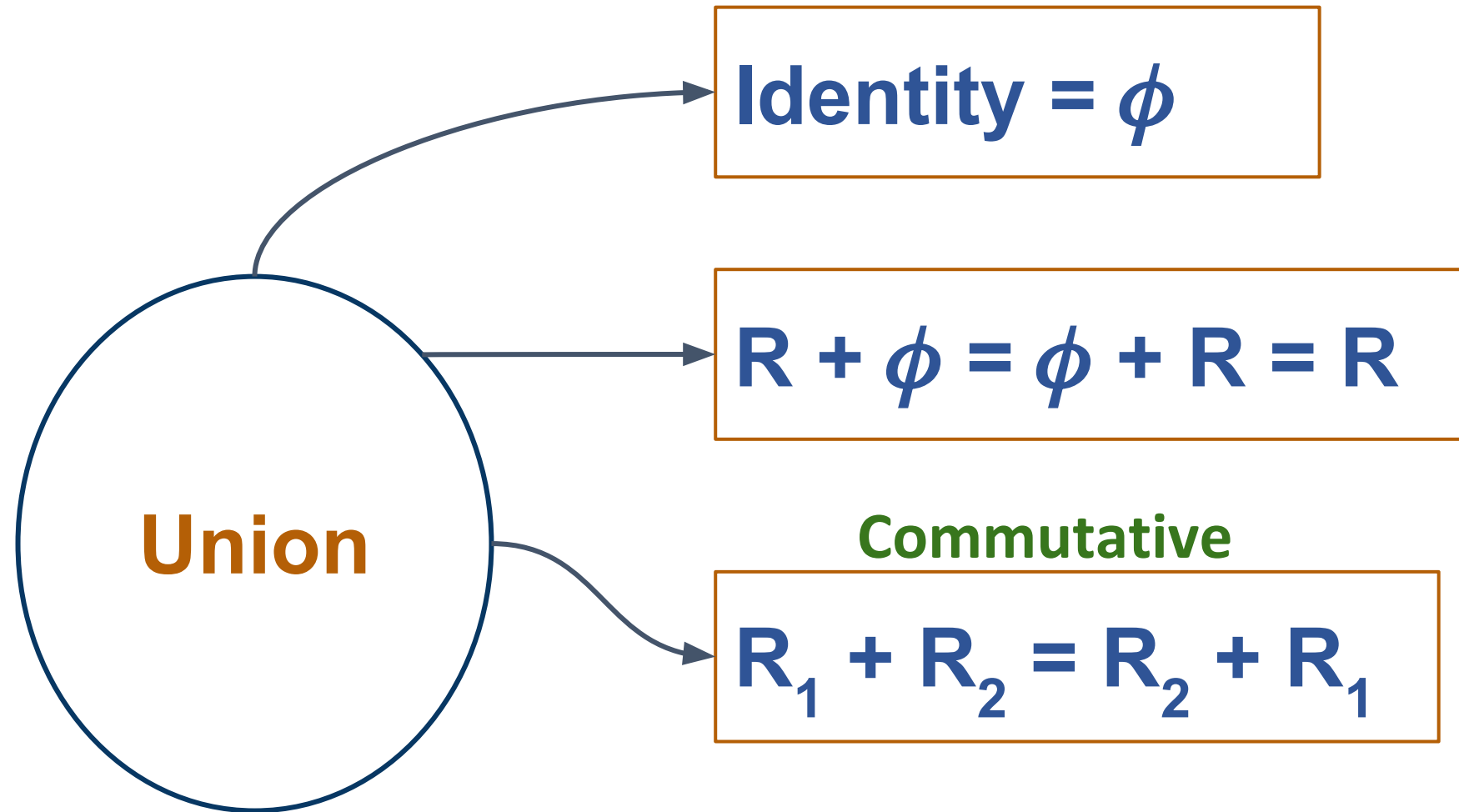Followed by 0 or more no. of b's

Ending with a 'c'

Example

$$(a \quad b^* \quad c)+ d$$

L = { ac,

abc,

abbb....c,

d }

Identity = ϕ

Union

Identity = ϕ

R + ϕ = ϕ + R = R

Union

Union

$$\text{Identity} = \phi$$

$$R + \phi = \phi + R = R$$

Commutative

$$R_1 + R_2 = R_2 + R_1$$

Union

$\text{Identity} = \phi$

$R + \phi = \phi + R = R$

Commutative

$R_1 + R_2 = R_2 + R_1$

Associative

$(R_1 + R_2) + R_3 = R_1 + (R_2 + R_3)$

Identity = ϵ or λ

Concatenation

Identity = ϵ or λ
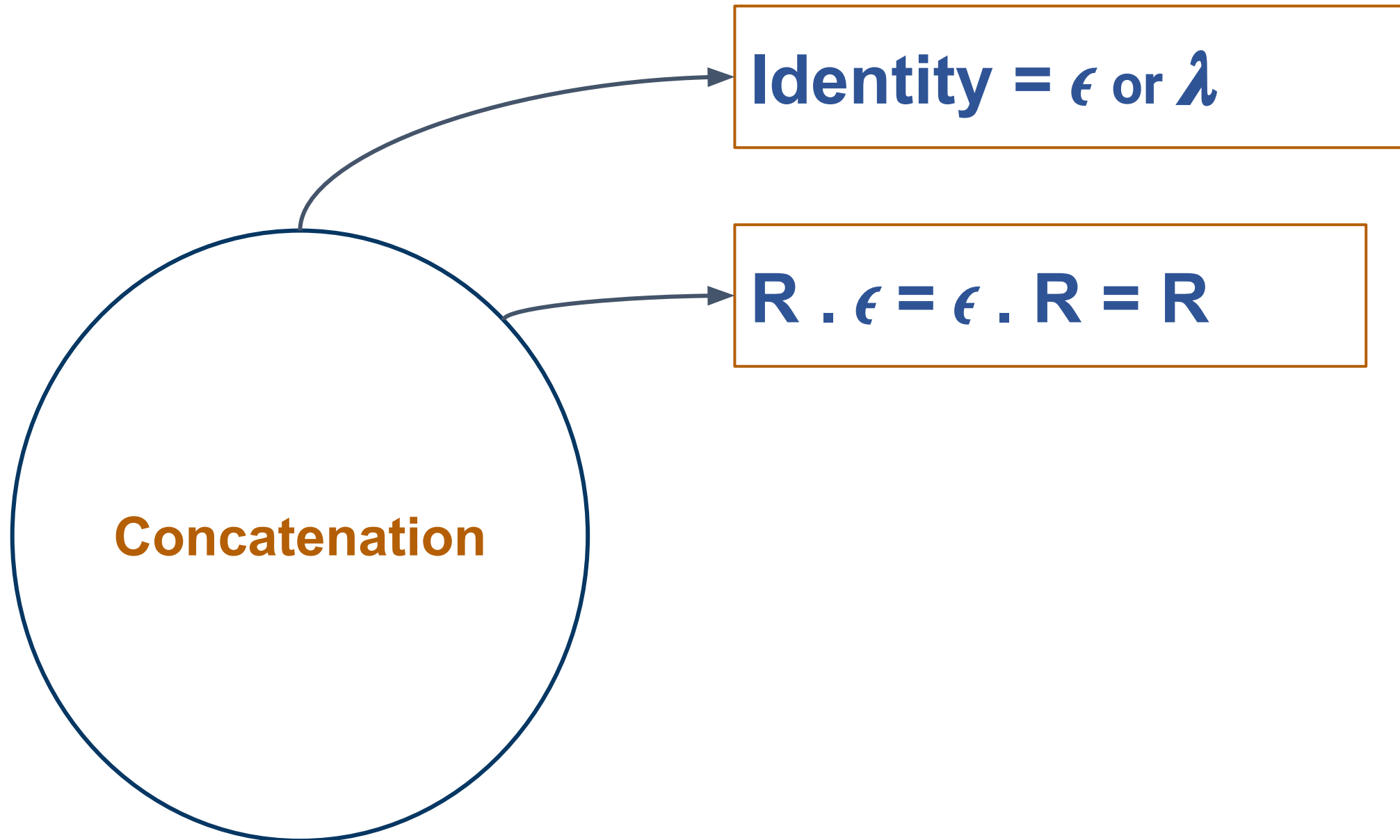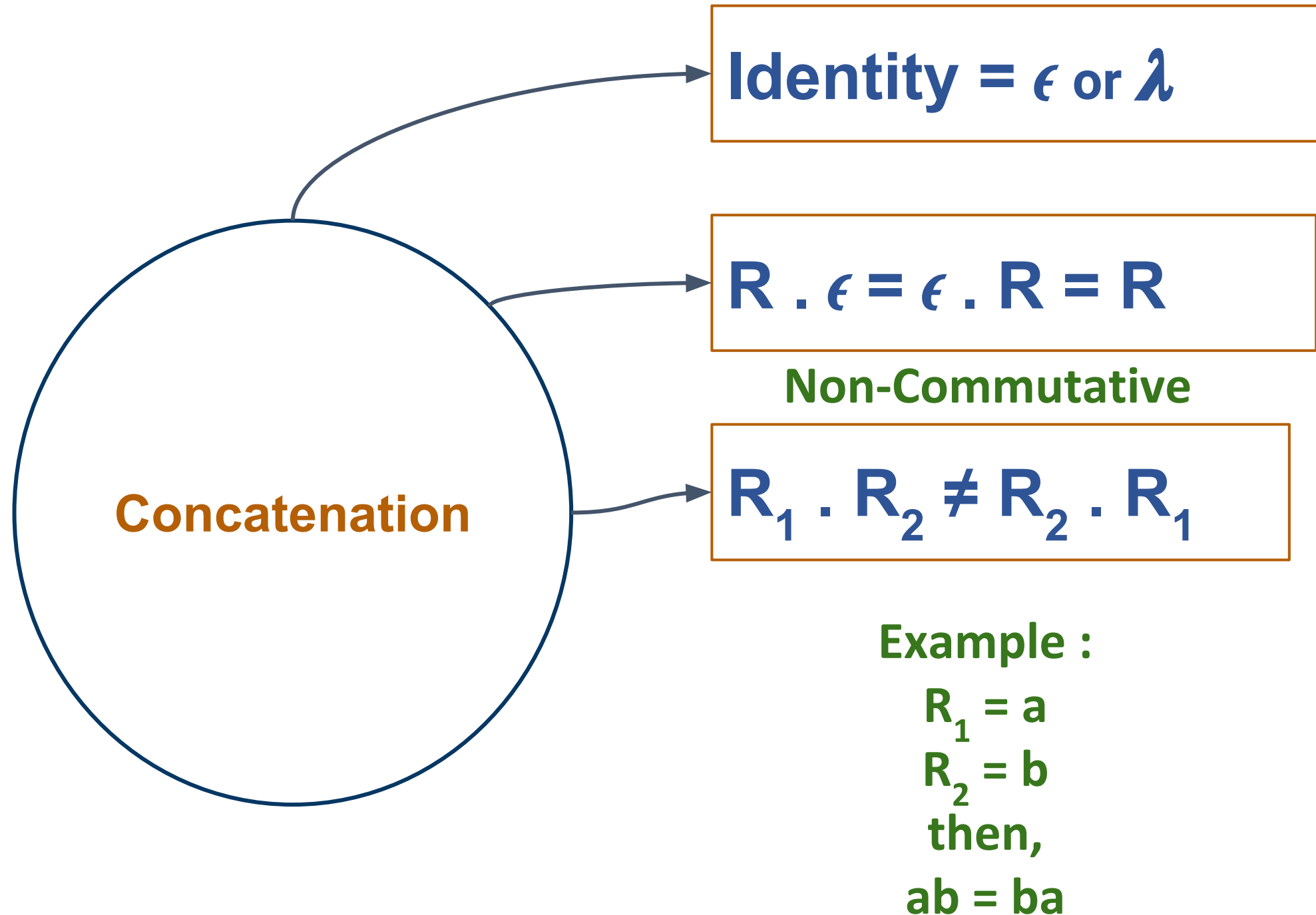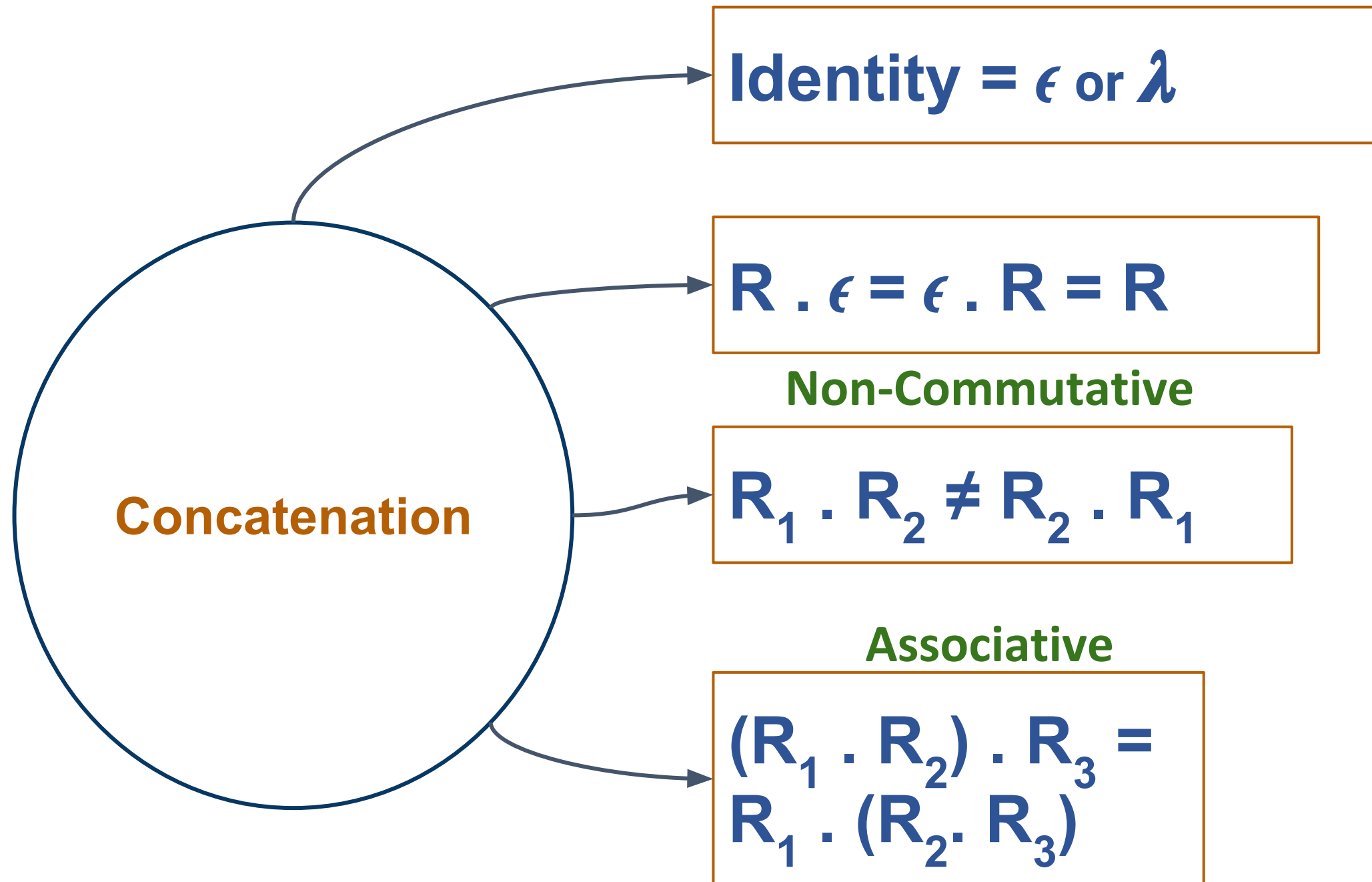
R . ϵ = ϵ . R = R
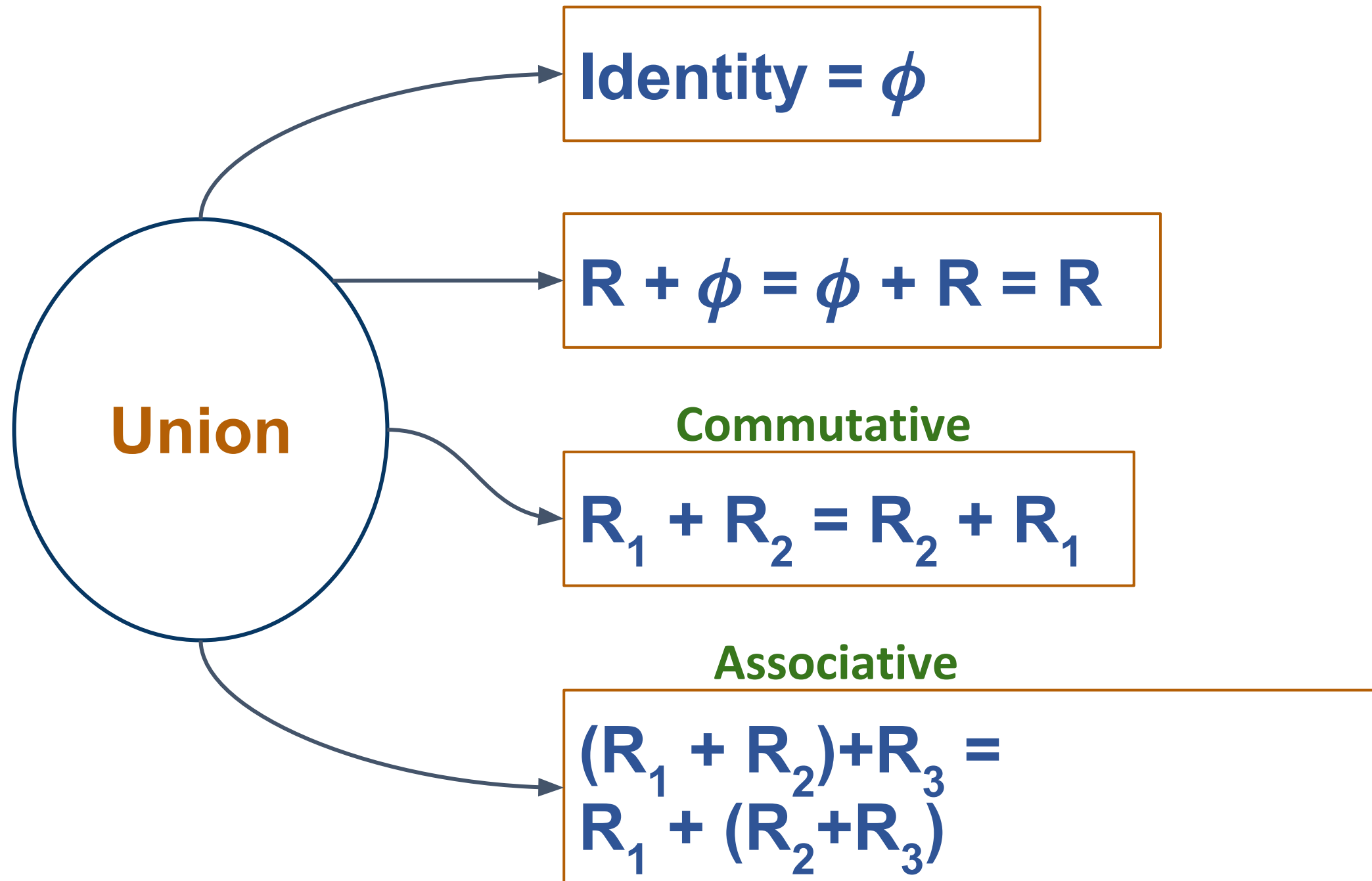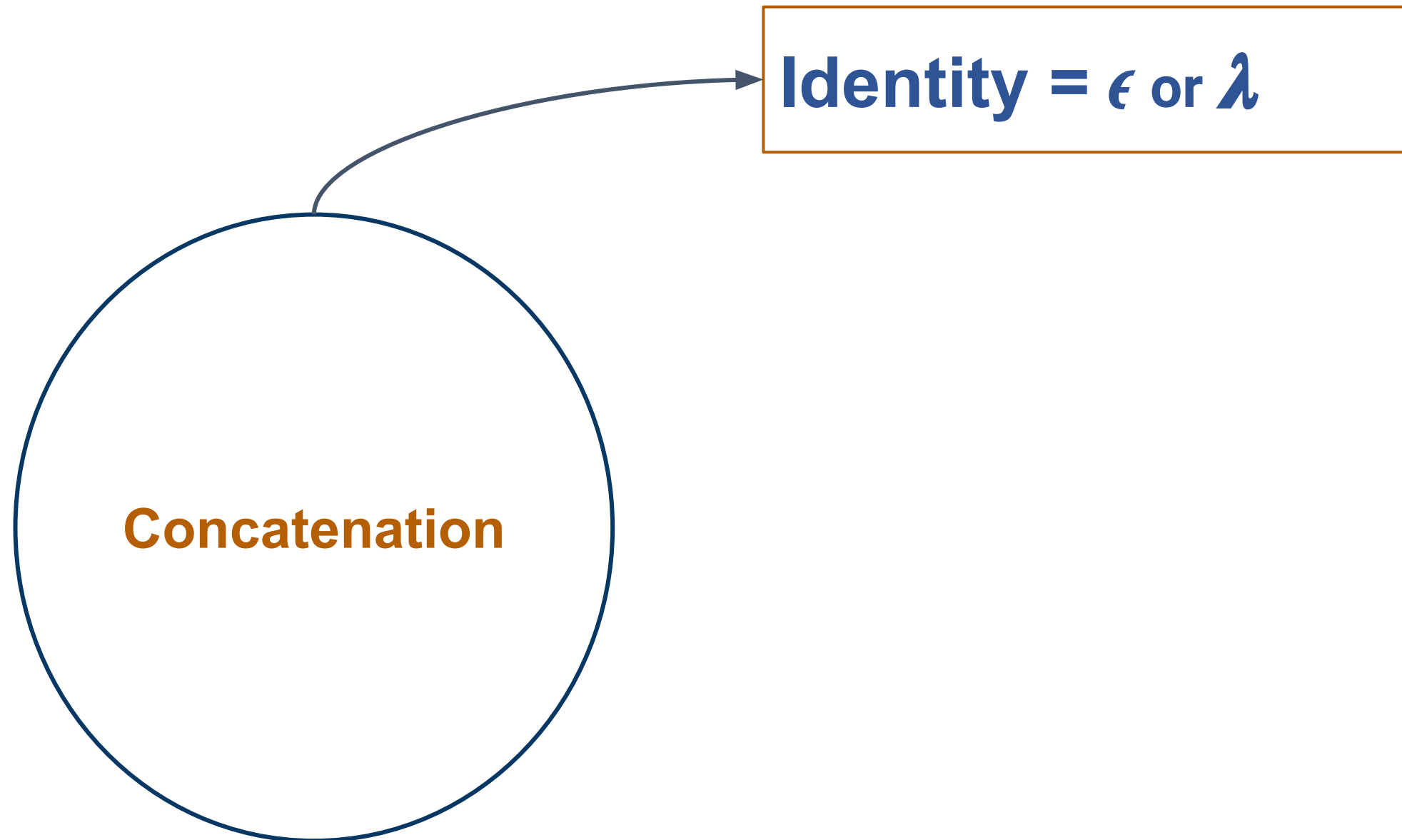
Concatenation

$$R . \phi = \phi . R = \phi$$

Let L_1, L_2 be languages, then the concatenation $L_1 \circ L_2 = \{w \mid w = xy, x \in L_1, y \in L_2\}$. If $L_2 = \emptyset$, then there is no string $y \in L_2$ and so there is no possible w such that $w = xy$. Thus for any L_1, we'll have $L_1 \circ \emptyset = \emptyset$.

Concatenation

Identity = ϵ or λ

R . ϵ = ϵ . R = R

Non-Commutative

$R_1 . R_2 \neq R_2 . R_1$

Example :
R_1 = a
R_2 = b
then,
ab = ba

Concatenation

Identity = ϵ or λ

R . ϵ = ϵ . R = R

Non-Commutative

$R_1 . R_2 \neq R_2 . R_1$

Associative

$(R_1 . R_2) . R_3 = R_1 . (R_2 . R_3)$

Closure

$$\epsilon* = \epsilon$$

$$\phi* = \epsilon$$

$$(R*)* = R*$$

$$(R + \epsilon)* = R*$$

$$R^+ = RR* = R*R$$

$$(R^+ + \epsilon) = R*$$

$$\epsilon + RR* = \epsilon + R^+ = R*$$

Closure

(a + b)*

= (a* + b*)*

= (a* b*)*

= (a + b*)*

= (a* + b)*

= a*(ba*)*

= b*(ab*)*

Construct a Regular Expression for a given language L

THANK YOU

Preet Kanwal

Department of Computer Science & Engineering

preetkanwal@pes.edu

+91 80 6666 3333 Extn 724