

Review of Math

MEB: Chapter 9



EECS3342 E: System
Specification and Refinement
Fall 2024

CHEN-WEI WANG

Learning Outcomes of this Lecture



This module is designed to help you review:

- Propositional Logic
- Predicate Logic
- Sets, Relations, and Functions

Propositional Logic (1)



- A **proposition** is a statement of claim that must be of either *true* or *false*, but not both.
- Basic logical operands are of type Boolean: *true* and *false*.
- We use logical operators to construct compound statements.
 - Unary logical operator: negation (\neg)

p	$\neg p$
<i>true</i>	<i>false</i>
<i>false</i>	<i>true</i>

- Binary logical operators: conjunction (\wedge), disjunction (\vee), implication (\Rightarrow), equivalence (\equiv), and if-and-only-if (\Leftrightarrow).

p	q	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$	$p \equiv q$
<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>
<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>
<i>false</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>false</i>
<i>false</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>

Propositional Logic: Implication (1)



- Written as $p \Rightarrow q$ [pronounced as “p implies q”]
 - We call p the antecedent, assumption, or premise.
 - We call q the consequence or conclusion.
- Compare the *truth* of $p \Rightarrow q$ to whether a contract is *honoured*:
 - antecedent/assumption/premise $p \approx$ promised terms [e.g., salary]
 - consequence/conclusion $q \approx$ obligations [e.g., duties]
- When the promised terms are met, then the contract is:
 - *honoured* if the obligations fulfilled. [$(\text{true} \Rightarrow \text{true}) \Leftrightarrow \text{true}$]
 - *breached* if the obligations violated. [$(\text{true} \Rightarrow \text{false}) \Leftrightarrow \text{false}$]
- When the promised terms are not met, then:
 - Fulfilling the obligation (q) or not ($\neg q$) does *not breach* the contract.

p	q	$p \Rightarrow q$
<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>false</i>	<i>true</i>

Propositional Logic: Implication (2)



There are alternative, equivalent ways to expressing $p \Rightarrow q$:

- **q if p**
q is *true* if p is *true*
- **p only if q**
If p is *true*, then for $p \Rightarrow q$ to be *true*, it can only be that q is also *true*.
Otherwise, if p is *true* but q is *false*, then $(\text{true} \Rightarrow \text{false}) \equiv \text{false}$.
- Note.** To prove $p \equiv q$, prove $p \iff q$ (pronounced: "p if and only if q"):
- **p if q** [$q \Rightarrow p$]
- **p only if q** [$p \Rightarrow q$]
- p is **sufficient** for q
For q to be *true*, it is sufficient to have p being *true*.
- q is **necessary** for p [similar to p only if q]
If p is *true*, then it is necessarily the case that q is also *true*.
Otherwise, if p is *true* but q is *false*, then $(\text{true} \Rightarrow \text{false}) \equiv \text{false}$.
- q **unless** $\neg p$ [When is $p \Rightarrow q$ *true*?]
If q is *true*, then $p \Rightarrow q$ *true* regardless of p.
If q is *false*, then $p \Rightarrow q$ cannot be *true* unless p is *false*.

5 of 41

Propositional Logic: Implication (3)



Given an implication $p \Rightarrow q$, we may construct its:

- **Inverse:** $\neg p \Rightarrow \neg q$ [negate antecedent and consequence]
- **Converse:** $q \Rightarrow p$ [swap antecedent and consequence]
- **Contrapositive:** $\neg q \Rightarrow \neg p$ [inverse of converse]

6 of 41

Propositional Logic (2)



- **Axiom:** Definition of \Rightarrow

$$p \Rightarrow q \equiv \neg p \vee q$$

- **Theorem:** Identity of \Rightarrow

$$\text{true} \Rightarrow p \equiv p$$

- **Theorem:** Zero of \Rightarrow

$$\text{false} \Rightarrow p \equiv \text{true}$$

- **Axiom:** De Morgan

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$

$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

- **Axiom:** Double Negation

$$p \equiv \neg(\neg p)$$

- **Theorem:** Contrapositive

$$p \Rightarrow q \equiv \neg q \Rightarrow \neg p$$

7 of 41

Predicate Logic (1)



- A **predicate** is a *universal* or *existential* statement about objects in some universe of discourse.
- Unlike propositions, predicates are typically specified using **variables**, each of which declared with some *range* of values.
- We use the following symbols for common numerical ranges:
 - \mathbb{Z} : the set of integers [$-\infty, \dots, -1, 0, 1, \dots, +\infty$]
 - \mathbb{N} : the set of natural numbers [$0, 1, \dots, +\infty$]
- Variable(s) in a predicate may be **quantified**:
 - **Universal quantification**:
All values that a variable may take satisfy certain property.
e.g., Given that i is a natural number, i is *always* non-negative.
 - **Existential quantification**:
Some value that a variable may take satisfies certain property.
e.g., Given that i is an integer, i *can be* negative.

8 of 41

Predicate Logic (2.1): Universal Q. (\forall)

- A **universal quantification** has the form $(\forall X \bullet R \Rightarrow P)$
 - X is a comma-separated list of variable names
 - R is a **constraint on types/ranges** of the listed variables
 - P is a **property** to be satisfied
- For all** (combinations of) values of variables listed in X that satisfies R , it is the case that P is satisfied.
 - $\forall i \bullet i \in \mathbb{N} \Rightarrow i \geq 0$ [true]
 - $\forall i \bullet i \in \mathbb{Z} \Rightarrow i \geq 0$ [false]
 - $\forall i, j \bullet i \in \mathbb{Z} \wedge j \in \mathbb{Z} \Rightarrow i < j \vee i > j$ [false]
- Proof Strategies**
 - How to prove $(\forall X \bullet R \Rightarrow P)$ **true**?
 - Hint.** When is $R \Rightarrow P$ **true**? [true \Rightarrow true, false \Rightarrow -]
 - Show that for all instances of $x \in X$ s.t. $R(x)$, $P(x)$ holds.
 - Show that for all instances of $x \in X$ it is the case $\neg R(x)$.
 - How to prove $(\forall X \bullet R \Rightarrow P)$ **false**?
 - Hint.** When is $R \Rightarrow P$ **false**? [true \Rightarrow false]
 - Give a **witness/counterexample** of $x \in X$ s.t. $R(x)$, $\neg P(x)$ holds.

9 of 41

Predicate Logic (2.2): Existential Q. (\exists)

- An **existential quantification** has the form $(\exists X \bullet R \wedge P)$
 - X is a comma-separated list of variable names
 - R is a **constraint on types/ranges** of the listed variables
 - P is a **property** to be satisfied
- There exist** (a combination of) values of variables listed in X that satisfy both R and P .
 - $\exists i \bullet i \in \mathbb{N} \wedge i \geq 0$ [true]
 - $\exists i \bullet i \in \mathbb{Z} \wedge i \geq 0$ [true]
 - $\exists i, j \bullet i \in \mathbb{Z} \wedge j \in \mathbb{Z} \wedge (i < j \vee i > j)$ [true]
- Proof Strategies**
 - How to prove $(\exists X \bullet R \wedge P)$ **true**?
 - Hint.** When is $R \wedge P$ **true**? [true \wedge true]
 - Give a **witness** of $x \in X$ s.t. $R(x)$, $P(x)$ holds.
 - How to prove $(\exists X \bullet R \wedge P)$ **false**?
 - Hint.** When is $R \wedge P$ **false**? [true \wedge false, false \wedge -]
 - Show that for all instances of $x \in X$ s.t. $R(x)$, $\neg P(x)$ holds.
 - Show that for all instances of $x \in X$ it is the case $\neg R(x)$.

10 of 41

Predicate Logic (3): Exercises

- Prove or disprove: $\forall x \bullet (x \in \mathbb{Z} \wedge 1 \leq x \leq 10) \Rightarrow x > 0$.
All 10 integers between 1 and 10 are greater than 0.
- Prove or disprove: $\forall x \bullet (x \in \mathbb{Z} \wedge 1 \leq x \leq 10) \Rightarrow x > 1$.
Integer 1 (a witness/counterexample) in the range between 1 and 10 is **not** greater than 1.
- Prove or disprove: $\exists x \bullet (x \in \mathbb{Z} \wedge 1 \leq x \leq 10) \wedge x > 1$.
Integer 2 (a witness) in the range between 1 and 10 is greater than 1.
- Prove or disprove that $\exists x \bullet (x \in \mathbb{Z} \wedge 1 \leq x \leq 10) \wedge x > 10$?
All integers in the range between 1 and 10 are **not** greater than 10.

11 of 41

Predicate Logic (4): Switching Quantifications

Conversions between \forall and \exists :

$$(\forall X \bullet R \Rightarrow P) \iff \neg(\exists X \bullet R \wedge \neg P)$$

$$(\exists X \bullet R \wedge P) \iff \neg(\forall X \bullet R \Rightarrow \neg P)$$

12 of 41

Sets: Definitions and Membership

- A **set** is a collection of objects.
 - Objects in a set are called its **elements** or **members**.
 - Order** in which elements are arranged does not matter.
 - An element can appear **at most once** in the set.
- We may define a set using:
 - Set Enumeration**: Explicitly list all members in a set.
e.g., $\{1, 3, 5, 7, 9\}$
 - Set Comprehension**: Implicitly specify the condition that all members satisfy.
e.g., $\{x \mid 1 \leq x \leq 10 \wedge x \text{ is an odd number}\}$
- An empty set (denoted as $\{\}$ or \emptyset) has no members.
- We may check if an element is a **member** of a set:
 - e.g., $5 \in \{1, 3, 5, 7, 9\}$ [true]
 - e.g., $4 \notin \{x \mid x \leq 1 \leq 10, x \text{ is an odd number}\}$ [true]
- The number of elements in a set is called its **cardinality**.
e.g., $|\emptyset| = 0$, $|\{x \mid x \leq 1 \leq 10, x \text{ is an odd number}\}| = 5$

13 of 41

Set Relations

Given two sets S_1 and S_2 :

- S_1 is a **subset** of S_2 if every member of S_1 is a member of S_2 .

$$S_1 \subseteq S_2 \iff (\forall x \bullet x \in S_1 \Rightarrow x \in S_2)$$

- S_1 and S_2 are **equal** iff they are the subset of each other.

$$S_1 = S_2 \iff S_1 \subseteq S_2 \wedge S_2 \subseteq S_1$$

- S_1 is a **proper subset** of S_2 if it is a strictly smaller subset.

$$S_1 \subset S_2 \iff S_1 \subseteq S_2 \wedge |S_1| < |S_2|$$

14 of 41

Set Relations: Exercises

$? \subseteq S$ always holds	[\emptyset and S]
$? \subset S$ always fails	[S]
$? \subset S$ holds for some S and fails for some S	[\emptyset]
$S_1 = S_2 \Rightarrow S_1 \subseteq S_2$?	[Yes]
$S_1 \subseteq S_2 \Rightarrow S_1 = S_2$?	[No]

15 of 41

Set Operations

Given two sets S_1 and S_2 :

- Union** of S_1 and S_2 is a set whose members are in either.

$$S_1 \cup S_2 = \{x \mid x \in S_1 \vee x \in S_2\}$$

- Intersection** of S_1 and S_2 is a set whose members are in both.

$$S_1 \cap S_2 = \{x \mid x \in S_1 \wedge x \in S_2\}$$

- Difference** of S_1 and S_2 is a set whose members are in S_1 but not S_2 .

$$S_1 \setminus S_2 = \{x \mid x \in S_1 \wedge x \notin S_2\}$$

16 of 41

Power Sets



The **power set** of a set S is a **set** of all S 's **subsets**.

$$\mathbb{P}(S) = \{s \mid s \subseteq S\}$$

The power set contains subsets of **cardinalities** $0, 1, 2, \dots, |S|$.
e.g., $\mathbb{P}(\{1, 2, 3\})$ is a set of sets, where each member set s has cardinality $0, 1, 2$, or 3 :

$$\left\{ \begin{array}{l} \emptyset, \\ \{1\}, \{2\}, \{3\}, \\ \{1, 2\}, \{2, 3\}, \{3, 1\}, \\ \{1, 2, 3\} \end{array} \right\}$$

Exercise: What is $\mathbb{P}(\{1, 2, 3, 4, 5\}) \setminus \mathbb{P}(\{1, 2, 3\})$?

17 of 41

Set of Tuples



Given n sets S_1, S_2, \dots, S_n , a **cross/Cartesian product** of these sets is a set of n -tuples.

Each **n -tuple** (e_1, e_2, \dots, e_n) contains n elements, each of which is a member of the corresponding set.

$$S_1 \times S_2 \times \dots \times S_n = \{(e_1, e_2, \dots, e_n) \mid e_i \in S_i \wedge 1 \leq i \leq n\}$$

e.g., $\{a, b\} \times \{2, 4\} \times \{\$, \&\}$ is a set of triples:

$$\begin{aligned} & \{a, b\} \times \{2, 4\} \times \{\$, \&\} \\ = & \{ (e_1, e_2, e_3) \mid e_1 \in \{a, b\} \wedge e_2 \in \{2, 4\} \wedge e_3 \in \{\$, \&\} \} \\ = & \left\{ \begin{array}{l} (a, 2, \$), (a, 2, \&), (a, 4, \$), (a, 4, \&), \\ (b, 2, \$), (b, 2, \&), (b, 4, \$), (b, 4, \&) \end{array} \right\} \end{aligned}$$

18 of 41

Relations (1): Constructing a Relation



A **relation** is a set of mappings, each being an **ordered pair** that maps a member of set S to a member of set T .

e.g., Say $S = \{1, 2, 3\}$ and $T = \{a, b\}$

- \emptyset is the **minimum** relation (i.e., an empty relation).
- $S \times T$ is the **maximum** relation (say r_1) between S and T , mapping from each member of S to each member in T :

$$\{(1, a), (1, b), (2, a), (2, b), (3, a), (3, b)\}$$

- $\{(x, y) \mid (x, y) \in S \times T \wedge x \neq 1\}$ is a relation (say r_2) that maps only some members in S to every member in T :

$$\{(2, a), (2, b), (3, a), (3, b)\}$$

19 of 41

Relations (2.1): Set of Possible Relations



- We use the **power set** operator to express the set of **all possible relations** on S and T :

$$\mathbb{P}(S \times T)$$

Each member in $\mathbb{P}(S \times T)$ is a relation.

- To declare a relation variable r , we use the colon ($:$) symbol to mean **set membership**:

$$r : \mathbb{P}(S \times T)$$

- Or alternatively, we write:

$$r : S \leftrightarrow T$$

where the set $S \leftrightarrow T$ is synonymous to the set $\mathbb{P}(S \times T)$

20 of 41

Relations (2.2): Exercise

Enumerate $\{a, b\} \leftrightarrow \{1, 2, 3\}$.

- **Hints:**

- You may enumerate all relations in $\mathbb{P}(\{a, b\} \times \{1, 2, 3\})$ via their **cardinalities**: $0, 1, \dots, |\{a, b\} \times \{1, 2, 3\}|$.
- What's the **maximum** relation in $\mathbb{P}(\{a, b\} \times \{1, 2, 3\})$?
 $\{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$
- The answer is a set containing **all** of the following relations:
 - Relation with cardinality 0: \emptyset
 - How many relations with cardinality 1? $\left[\binom{|\{a, b\} \times \{1, 2, 3\}|}{1} = 6 \right]$
 - How many relations with cardinality 2? $\left[\binom{|\{a, b\} \times \{1, 2, 3\}|}{2} = \frac{6 \times 5}{2!} = 15 \right]$

...

- Relation with cardinality $|\{a, b\} \times \{1, 2, 3\}|$:
 $\{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$

21 of 41

Relations (3.1): Domain, Range, Inverse

Given a relation

$$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$$

- **domain** of r : set of first-elements from r
 - Definition: $\text{dom}(r) = \{d \mid (d, r') \in r\}$
 - e.g., $\text{dom}(r) = \{a, b, c, d, e, f\}$
 - ASCII syntax: `dom(r)`
- **range** of r : set of second-elements from r
 - Definition: $\text{ran}(r) = \{r' \mid (d, r') \in r\}$
 - e.g., $\text{ran}(r) = \{1, 2, 3, 4, 5, 6\}$
 - ASCII syntax: `ran(r)`
- **inverse** of r : a relation like r with elements swapped
 - Definition: $r^{-1} = \{(r', d) \mid (d, r') \in r\}$
 - e.g., $r^{-1} = \{(1, a), (2, b), (3, c), (4, a), (5, b), (6, c), (1, d), (2, e), (3, f)\}$
 - ASCII syntax: `r~`

22 of 41

Relations (3.2): Image

Given a relation

$$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$$

relational image of r over set s : sub-range of r mapped by s .

- Definition: $r[s] = \{r' \mid (d, r') \in r \wedge d \in s\}$
- e.g., $r[\{a, b\}] = \{1, 2, 4, 5\}$
- ASCII syntax: `r[s]`

23 of 41

Relations (3.3): Restrictions

Given a relation

$$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$$

- **domain restriction** of r over set ds : sub-relation of r with domain ds .
 - Definition: $ds \triangleleft r = \{(d, r') \mid (d, r') \in r \wedge d \in ds\}$
 - e.g., $\{a, b\} \triangleleft r = \{(a, 1), (b, 2), (a, 4), (b, 5)\}$
 - ASCII syntax: `ds <| r`
- **range restriction** of r over set rs : sub-relation of r with range rs .
 - Definition: $r \triangleright rs = \{(d, r') \mid (d, r') \in r \wedge r' \in rs\}$
 - e.g., $r \triangleright \{1, 2\} = \{(a, 1), (b, 2), (d, 1), (e, 2)\}$
 - ASCII syntax: `r |> rs`

24 of 41

Relations (3.4): Subtractions



Given a relation

$$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$$

- **domain subtraction** of r over set ds : sub-relation of r with domain not ds .
 - Definition: $ds \triangleleft r = \{(d, r') \mid (d, r') \in r \wedge d \notin ds\}$
 - e.g., $\{a, b\} \triangleleft r = \{(c, 3), (c, 6), (d, 1), (e, 2), (f, 3)\}$
 - ASCII syntax: `ds <<| r`
- **range subtraction** of r over set rs : sub-relation of r with range not rs .
 - Definition: $r \triangleright rs = \{(d, r') \mid (d, r') \in r \wedge r' \notin rs\}$
 - e.g., $r \triangleright \{1, 2\} = \{(c, 3), (a, 4), (b, 5), (c, 6), (f, 3)\}$
 - ASCII syntax: `r |>> rs`

25 of 41

Relations (3.5): Overriding



Given a relation

$$r = \{(a, 1), (b, 2), (c, 3), (a, 4), (b, 5), (c, 6), (d, 1), (e, 2), (f, 3)\}$$

overriding of r with relation t : a relation which agrees with t within $\text{dom}(t)$, and agrees with r outside $\text{dom}(t)$

- Definition: $r \triangleleft t = \{(d, r') \mid (d, r') \in t \vee ((d, r') \in r \wedge d \notin \text{dom}(t))\}$
- e.g.,

$$r \triangleleft t = \{(a, 3), (c, 4)\} \cup \{(b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}$$

$$= \{(d, r') \mid (d, r') \in t\} \cup \{(d, r') \mid (d, r') \in r \wedge d \notin \text{dom}(t)\}$$

$$= \{(a, 3), (c, 4), (b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}$$
- ASCII syntax: `r <+ t`

26 of 41

Relations (4): Exercises



1. Define $r[s]$ in terms of other relational operations.

Answer: $r[s] = \text{ran}(s \triangleleft r)$

e.g.,

$$r[\underbrace{\{a, b\}}_s] = \text{ran}(\underbrace{\{(a, 1), (b, 2), (a, 4), (b, 5)\}}_{\{a, b\} \triangleleft r}) = \{1, 2, 4, 5\}$$

2. Define $r \triangleleft t$ in terms of other relational operators.

Answer: $r \triangleleft t = t \cup (\text{dom}(t) \triangleleft r)$

e.g.,

$$\begin{aligned} r \triangleleft t &= \{(a, 3), (c, 4)\} \\ &= \underbrace{\{(a, 3), (c, 4)\}}_t \cup \underbrace{\{(b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\}}_{\text{dom}(t) \triangleleft r} \\ &= \{(a, 3), (c, 4), (b, 2), (b, 5), (d, 1), (e, 2), (f, 3)\} \end{aligned}$$

27 of 41

Functions (1): Functional Property



- A **relation** r on sets S and T (i.e., $r \in S \leftrightarrow T$) is also a **function** if it satisfies the **functional property**:
 $\text{isFunction}(r)$

\iff

$$\forall s, t_1, t_2 \bullet (s \in S \wedge t_1 \in T \wedge t_2 \in T) \Rightarrow ((s, t_1) \in r \wedge (s, t_2) \in r \Rightarrow t_1 = t_2)$$

- That is, in a **function**, it is forbidden for a member of S to map to more than one members of T .
- Equivalently, in a **function**, two distinct members of T cannot be mapped by the same member of S .
- e.g., Say $S = \{1, 2, 3\}$ and $T = \{a, b\}$, which of the following **relations** satisfy the above **functional property**?
 - $S \times T$ [No]
Witness 1: $(1, a), (1, b)$; **Witness 2:** $(2, a), (2, b)$; **Witness 3:** $(3, a), (3, b)$.
 - $(S \times T) \setminus \{(x, y) \mid (x, y) \in S \times T \wedge x = 1\}$ [No]
Witness 1: $(2, a), (2, b)$; **Witness 2:** $(3, a), (3, b)$
 - $\{(1, a), (2, b), (3, a)\}$ [Yes]
 - $\{(1, a), (2, b)\}$ [Yes]

28 of 41

Functions (2.1): Total vs. Partial

Given a relation $r \in S \leftrightarrow T$

- r is a **partial function** if it satisfies the **functional property**:

$$r \in S \rightharpoonup T \iff (\text{isFunctional}(r) \wedge \text{dom}(r) \subseteq S)$$

Remark. $r \in S \rightharpoonup T$ means there may (or may not) be $s \in S$ s.t. $r(s)$ is **undefined**.

- e.g., $\{ \{(2, a), (1, b)\}, \{(2, a), (3, a), (1, b)\} \} \subseteq \{1, 2, 3\} \rightharpoonup \{a, b\}$
- ASCII syntax: $r : +->$
- r is a **total function** if there is a mapping for each $s \in S$:

$$r \in S \rightarrow T \iff (\text{isFunctional}(r) \wedge \text{dom}(r) = S)$$

Remark. $r \in S \rightarrow T$ implies $r \in S \rightharpoonup T$, but not vice versa. Why?

- e.g., $\{(2, a), (3, a), (1, b)\} \in \{1, 2, 3\} \rightarrow \{a, b\}$
- e.g., $\{(2, a), (1, b)\} \notin \{1, 2, 3\} \rightarrow \{a, b\}$
- ASCII syntax: $r : -->$

29 of 41

Functions (2.2): Relation Image vs. Function Application

- Recall: A **function** is a **relation**, but a **relation** is not necessarily a **function**.
- Say we have a **partial function** $f \in \{1, 2, 3\} \rightharpoonup \{a, b\}$:

$$f = \{(3, a), (1, b)\}$$

- With f wearing the **relation** hat, we can invoke **relational images**:

$$\begin{aligned} f[\{3\}] &= \{a\} \\ f[\{1\}] &= \{b\} \\ f[\{2\}] &= \emptyset \end{aligned}$$

Remark. Given that the inputs are **singleton** sets (e.g., $\{3\}$), so are the output sets (e.g., $\{a\}$). \therefore Each member in the domain is mapped to at most one member in the range.

- With f wearing the **function** hat, we can invoke **functional applications**:

$$\begin{aligned} f(3) &= a \\ f(1) &= b \\ f(2) &\text{ is } \text{undefined} \end{aligned}$$

30 of 41

Functions (2.3): Modelling Decision

An organization has a system for keeping **track** of its employees as to where they are on the premises (e.g., ``Zone A, Floor 23``). To achieve this, each employee is issued with an active badge which, when scanned, synchronizes their current positions to a central database.

Assume the following two sets:

- Employee** denotes the **set** of all employees working for the organization.
- Location** denotes the **set** of all valid locations in the organization.

- Is it appropriate to **model/formalize** such a **track** functionality as a **relation** (i.e., **where.is** \in **Employee** \leftrightarrow **Location**)?

Answer. No – an employee cannot be at distinct locations simultaneously.

e.g., $\text{where_is}[\text{Alan}] = \{ \text{``Zone A, Floor 23``}, \text{``Zone C, Floor 46``} \}$

- How about a **total function** (i.e., **where.is** \in **Employee** \rightarrow **Location**)?

Answer. No – in reality, not necessarily all employees show up.

e.g., $\text{where_is}(\text{Mark})$ should be **undefined** if Mark happens to be on vacation.

- How about a **partial function** (i.e., **where.is** \in **Employee** \rightharpoonup **Location**)?

Answer. Yes – this addresses the inflexibility of the total function.

31 of 41

Functions (3.1): Injective Functions

Given a **function** f (either **partial** or **total**):

- f is **injective/one-to-one/an injection** if f does not map more than one members of S to a single member of T .

$$\text{isInjective}(f)$$

$$\iff$$

$$\forall s_1, s_2, t \bullet (s_1 \in S \wedge s_2 \in S \wedge t \in T) \Rightarrow ((s_1, t) \in f \wedge (s_2, t) \in f \Rightarrow s_1 = s_2)$$

- If f is a **partial injection**, we write: $f \in S \rightharpoonup T$

- e.g., $\{\emptyset, \{(1, a)\}, \{(2, a), (3, b)\}\} \subseteq \{1, 2, 3\} \rightharpoonup \{a, b\}$

- e.g., $\{(1, b), (2, a), (3, b)\} \notin \{1, 2, 3\} \rightharpoonup \{a, b\}$

[total, not inj.]

- e.g., $\{(1, b), (3, b)\} \notin \{1, 2, 3\} \rightharpoonup \{a, b\}$

[partial, not inj.]

- ASCII syntax: $f : >+>$

- If f is a **total injection**, we write: $f \in S \rightarrow T$

- e.g., $\{1, 2, 3\} \rightarrow \{a, b\} = \emptyset$

- e.g., $\{(2, d), (1, a), (3, c)\} \in \{1, 2, 3\} \rightarrow \{a, b, c, d\}$

- e.g., $\{(2, d), (1, c)\} \notin \{1, 2, 3\} \rightarrow \{a, b, c, d\}$

[not total, inj.]

- e.g., $\{(2, d), (1, c), (3, d)\} \notin \{1, 2, 3\} \rightarrow \{a, b, c, d\}$

[total, not inj.]

- ASCII syntax: $f : >->$

32 of 41

Functions (3.2): Surjective Functions



Given a **function** f (either partial or total):

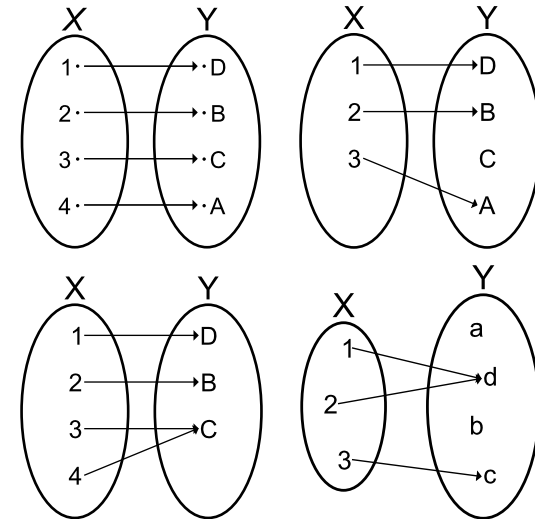
- f is **surjective/onto/a surjection** if f maps to all members of T .

$$\text{isSurjective}(f) \iff \text{ran}(f) = T$$

- If f is a **partial surjection**, we write: $f \in S \twoheadrightarrow T$
 - e.g., $\{ \{(1, \mathbf{b}), (2, \mathbf{a})\}, \{(1, \mathbf{b}), (2, \mathbf{a}), (3, \mathbf{b})\} \subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$
 - e.g., $\{(2, \mathbf{a}), (1, \mathbf{a}), (3, \mathbf{a})\} \not\subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$ [total, not sur.]
 - e.g., $\{(2, \mathbf{b}), (1, \mathbf{b})\} \not\subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$ [partial, not sur.]
 - ASCII syntax: $f : ++>>$
- If f is a **total surjection**, we write: $f \in S \twoheadrightarrow T$
 - e.g., $\{ \{(2, \mathbf{a}), (1, \mathbf{b}), (3, \mathbf{a})\}, \{(2, \mathbf{b}), (1, \mathbf{a}), (3, \mathbf{b})\} \subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$
 - e.g., $\{(2, \mathbf{a}), (3, \mathbf{b})\} \not\subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$ [not total, sur.]
 - e.g., $\{(2, \mathbf{a}), (3, \mathbf{a}), (1, \mathbf{a})\} \not\subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b\}$ [total., not sur.]
 - ASCII syntax: $f : -->>$

33 of 41

Functions (4.1): Exercises



35 of 41

Functions (3.3): Bijective Functions



Given a function f :

f is **bijective/a bijection/one-to-one correspondence** if f is **total**, **injective**, and **surjective**.

- e.g., $\{1, 2, 3\} \twoheadrightarrow \{a, b\} = \emptyset$
- e.g., $\{ \{(1, \mathbf{a}), (2, \mathbf{b}), (3, \mathbf{c})\}, \{(2, \mathbf{a}), (3, \mathbf{b}), (1, \mathbf{c})\} \subseteq \{1, 2, 3\} \twoheadrightarrow \{a, b, c\}$
- e.g., $\{(2, \mathbf{b}), (3, \mathbf{c}), (4, \mathbf{a})\} \not\subseteq \{1, 2, 3, 4\} \twoheadrightarrow \{a, b, c\}$ [not total, inj., sur.]
- e.g., $\{(1, \mathbf{a}), (2, \mathbf{b}), (3, \mathbf{c}), (4, \mathbf{a})\} \not\subseteq \{1, 2, 3, 4\} \twoheadrightarrow \{a, b, c\}$ [total, not inj., sur.]
- e.g., $\{(1, \mathbf{a}), (2, \mathbf{c})\} \not\subseteq \{1, 2\} \twoheadrightarrow \{a, b, c\}$ [total, inj., not sur.]
- ASCII syntax: $f : >->>$

34 of 41

Functions (4.2): Modelling Decisions



- Should an array a declared as "String[] a " be **modelled/formalized** as a **partial** function (i.e., $a \in \mathbb{Z} \twoheadrightarrow \text{String}$) or a **total** function (i.e., $a \in \mathbb{Z} \rightarrow \text{String}$)?
Answer. $a \in \mathbb{Z} \rightarrow \text{String}$ is not appropriate as:
 - Indices are non-negative (i.e., $a(i)$, where $i < 0$, is **undefined**).
 - Each array size is finite: not all positive integers are valid indices.
- What does it mean if an **array** is **modelled/formalized** as a **partial injection** (i.e., $a \in \mathbb{Z} \twoheadrightarrow \text{String}$)?
Answer. It means that the array does not contain any duplicates.
- Can an integer array "int[] a " be **modelled/formalized** as a **partial surjection** (i.e., $a \in \mathbb{Z} \twoheadrightarrow \mathbb{Z}$)?
Answer. Yes, if a stores all 2^{32} integers (i.e., $[-2^{31}, 2^{31} - 1]$).
- Can a string array "String[] a " be **modelled/formalized** as a **partial surjection** (i.e., $a \in \mathbb{Z} \twoheadrightarrow \text{String}$)?
Answer. No \because # possible strings is ∞ .
- Can an integer array "int[]" storing all 2^{32} values be **modelled/formalized** as a **bijection** (i.e., $a \in \mathbb{Z} \twoheadrightarrow \mathbb{Z}$)?

Answer. No, because it cannot be **total** (as discussed earlier).

36 of 41

Beyond this lecture ...



- For the $where_is \in Employee \rightarrow Location$ model, what does it mean when it is:
 - Injective** [$where_is \in Employee \rightarrow Location$]
 - Surjective** [$where_is \in Employee \rightarrow Location$]
 - Bijjective** [$where_is \in Employee \rightarrow Location$]
- Review examples discussed in your earlier math courses on **logic** and **set theory**.

37 of 41

Index (1)



Learning Outcomes of this Lecture

Propositional Logic (1)

Propositional Logic: Implication (1)

Propositional Logic: Implication (2)

Propositional Logic: Implication (3)

Propositional Logic (2)

Predicate Logic (1)

Predicate Logic (2.1): Universal Q. (\forall)

Predicate Logic (2.2): Existential Q. (\exists)

Predicate Logic (3): Exercises

Predicate Logic (4): Switching Quantifications

38 of 41

Index (2)



Sets: Definitions and Membership

Set Relations

Set Relations: Exercises

Set Operations

Power Sets

Set of Tuples

Relations (1): Constructing a Relation

Relations (2.1): Set of Possible Relations

Relations (2.2): Exercise

Relations (3.1): Domain, Range, Inverse

Relations (3.2): Image

39 of 41

Index (3)



Relations (3.3): Restrictions

Relations (3.4): Subtractions

Relations (3.5): Overriding

Relations (4): Exercises

Functions (1): Functional Property

Functions (2.1): Total vs. Partial

Functions (2.2):

Relation Image vs. Function Application

Functions (2.3): Modelling Decision

Functions (3.1): Injective Functions

Functions (3.2): Surjective Functions

40 of 41

Index (4)



[Functions \(3.3\): Bijective Functions](#)

[Functions \(4.1\): Exercises](#)

[Functions \(4.2\): Modelling Decisions](#)

[Beyond this lecture ...](#)