Binary Relations
Part One
Outline for Today

- **Binary Relations**
  - Reasoning about connections between *pairs of* objects

- **Equivalence Relations**
  - Reasoning about clusters
Relationships

• In CS103, you've seen examples of relationships
  • between sets:
    \[ A \subseteq B \]
  • between numbers:
    \[ x < y \quad x \equiv_k y \quad x \leq y \]
  • between people:
    \[ p \text{ loves } q \]

• Since these relations focus on connections between two objects, they are called **binary relations**.
  • The “binary” here means “pertaining to two things,” not like computer code in 0s and 1s
What exactly is a binary relation?
$10 < 12$
5 \lessdot -2
7 ≡₃ 10
6 ≡₃ 11
$aRb$
$aRb$
Binary Relations

- A binary relation over a set $A$ is a predicate $R$ that can be applied to pairs of elements drawn from $A$.
- If $R$ is a binary relation over $A$ and it holds for the pair $(a, b)$, we write $aRb$.
  - $3 = 3$  \hspace{1cm}  $5 < 7$  \hspace{1cm}  $\emptyset \subseteq \mathbb{N}$
- If $R$ is a binary relation over $A$ and it does not hold for the pair $(a, b)$, we write $aRb$.
  - $4 \neq 3$  \hspace{1cm}  $4 \preceq 3$  \hspace{1cm}  $\mathbb{N} \nsubseteq \emptyset$
Properties of Relations

• Generally speaking, if $R$ is a binary relation over a set $A$, the order of the operands is significant.
  
  • For example, $3 < 5$, but $5 \not< 3$.
  
  • There are some specific relations (e.g., equals) for which order is irrelevant—more on this later!

• Relations are always defined relative to some underlying set.
  
  • It's not meaningful to ask whether $☺ \subseteq 15$, for example, since $\subseteq$ is defined over sets, not arbitrary objects.
We can visualize a binary relation $R$ over a set $A$ by drawing the elements of $A$ and drawing a line between an element $a$ and an element $b$ if $aRb$ is true.

Example: the relation $a \mid b$ (meaning “$a$ divides $b$”) over the set $\{1, 2, 3, 4\}$ looks like this:
We can visualize a binary relation $R$ over a set $A$ by drawing the elements of $A$ and drawing a line between an element $a$ and an element $b$ if $aRb$ is true.

Example: the relation $a \neq b$ over the set $\{1, 2, 3, 4\}$ looks like this:
We can visualize a binary relation $R$ over a set $A$ by drawing the elements of $A$ and drawing a line between an element $a$ and an element $b$ if $aRb$ is true.

Example: the relation $a = b$ over the set $\{1, 2, 3, 4\}$ looks like this:
Visualizing Relations

• We can visualize a binary relation $R$ over a set $A$ by drawing the elements of $A$ and drawing a line between an element $a$ and an element $b$ if $aRb$ is true.

• Example: below is some relation over $\{1, 2, 3, 4\}$ that's a totally valid relation even though there doesn't appear to be a simple unifying rule.
Capturing Structure
Capturing Structure

- Binary relations are an excellent way for capturing certain structures that appear in computer science.
- Today, we'll look at one of them (partitions).
- Along the way, we'll explore how to write proofs about definitions given in first-order logic.
Partitions
Partitions

• A *partition of a set* is a way of splitting the set into disjoint, nonempty subsets so that every element belongs to exactly one subset.
  • Two sets are *disjoint* if their intersection is the empty set; formally, sets $S$ and $T$ are disjoint if $S \cap T = \emptyset$.

• Intuitively, a partition of a set breaks the set apart into smaller pieces.

• There doesn't have to be any rhyme or reason to what those pieces are, though often there is one.
What's the connection between partitions and binary relations?
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.

\[ aRa \]

\[ aRb \rightarrow bRa \]

\[ aRb \land bRc \rightarrow aRc \]
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.

∀a ∈ A. aRa

∀a ∈ A. ∀b ∈ A. (aRb → bRa)

∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
\begin{align*}
\forall a \in A. \ aRa \\
\forall a \in A. \ \forall b \in A. \ (aRb \to bRa) \\
\forall a \in A. \ \forall b \in A. \ \forall c \in A. \ (aRb \land bRc \to aRc)
\end{align*}
Reflexivity

• In some relations, it happens that every element in the set relates to itself.

• Examples:
  • $x = x$ for any $x$.
  • $A \subseteq A$ for any set $A$.
  • $x \equiv_k x$ for any $x$.

• Relations of this sort are called reflexive.

• Formally speaking, a binary relation $R$ over a set $A$ is reflexive if the following first-order statement is true:

  $$\forall a \in A. \ aRa$$
Reflexivity Visualized

\[ \forall a \in A. \ aRa \]
\( \forall a \in A. aRa \)

Let \( R \) be the binary relation given by the drawing to the left. How many of the following objects are reflexive?

\[ R, \smiley, \smiley, \sad, \smiley, \smiley, \sad, \sad, \smiley, \smiley, \]

Answer at PollEv.com/cs103 or text CS103 to 22333 once to join, then 0, 1, 2, 3, 4, or 5.
∀a ∈ A. aRa
\( \forall a \in A. \ aRa \)
∀a ∈ A. aRa
This means that $R$ is not reflexive, since the first-order logic statement given below is not true.

\[ \forall a \in A. \, aRa \]
Is reflexive?
Is reflexive?

∀a ∈ ??.

\[ a \overset{??}{\to} a \]
Reflexivity is a property of relations, not individual objects.
\[ \forall a \in A. \ aRa \]

\[ \forall a \in A. \ \forall b \in A. \ (aRb \to bRa) \]

\[ \forall a \in A. \ \forall b \in A. \ \forall c \in A. \ (aRb \land bRc \to aRc) \]
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.

\[ \forall a \in A. \ aRa \]

\[ \forall a \in A. \ \forall b \in A. \ (aRb \rightarrow bRa) \]

\[ \forall a \in A. \ \forall b \in A. \ \forall c \in A. \ (aRb \land bRc \rightarrow a Rc) \]
Symmetry

- In some relations, the relative order of the objects doesn't matter. In other words, *if a is related to b, then b is related to a.*

- Examples:
  - If \( x = y \), then \( y = x \).
  - If \( x \equiv_k y \), then \( y \equiv_k x \).

- These relations are called **symmetric**.

- Formally: a binary relation \( R \) over a set \( A \) is called symmetric if the following first-order statement is true about \( R \):

\[
\forall a \in A. \forall b \in A. (aRb \rightarrow bRa)
\]
∀a ∈ A. ∀b ∈ A. (aRb → bRa)
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

∀a ∈ A. ∀b ∈ A. (aRb → bRa)
∀a ∈ A. ∀b ∈ A. (aRb → bRa)

Is this relation symmetric?
Is This Relation Symmetric?

∀a ∈ A. ∀b ∈ A. (aRb → bRa)
Is This Relation Symmetric?

∀a ∈ A. ∀b ∈ A. (aRb → bRa)
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

∀a ∈ A. ∀b ∈ A. (aRb → bRa)
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. (aRb \rightarrow bRa) \]
Is This Relation Symmetric?

\[
\forall a \in A. \forall b \in A. (aRb \rightarrow bRa)
\]
Is This Relation Symmetric?

\[ \forall a \in A. \forall b \in A. \ (aRb \rightarrow bRa) \]
∀a ∈ A. aRa

∀a ∈ A. ∀b ∈ A. (aRb → bRa)

∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
Relation this person holds: “Are these two things in the same partition?” for some mystery partition.

\[ \forall a \in A. \ aRa \]

\[ \forall a \in A. \ \forall b \in A. \ (aRb \rightarrow bRa) \]

\[ \forall a \in A. \ \forall b \in A. \ \forall c \in A. \ (aRb \land bRc \rightarrow aRc) \]
Transitivity

• In some relations, it happens that whenever \( a \) is related to \( b \) and \( b \) is related to \( c \), we know \( a \) is related to \( c \).

• Examples:
  • If \( x = y \) and \( y = z \), then \( x = z \).
  • If \( R \subseteq S \) and \( S \subseteq T \), then \( R \subseteq T \).
  • If \( x \equiv_k y \) and \( y \equiv_k z \), then \( x \equiv_k z \).

• These relations are called **transitive**.

• A binary relation \( R \) over a set \( A \) is called **transitive** if the following first-order statement is true about \( R \):

\[
\forall a \in A. \forall b \in A. \forall c \in A. (aRb \land bRc \rightarrow aRc)
\]
Transitivity Visualized

\[ \forall a \in A. \forall b \in A. \forall c \in A. (aRb \land bRc \rightarrow aRc) \]
Is This Relation Transitive?

∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
Is This Relation Transitive?

$$\forall a \in A. \forall b \in A. \forall c \in A. (aRb \land bRc \rightarrow aRc)$$
Is This Relation Transitive?

∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
Is This Relation Transitive?

∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
Equivalence Relations

- An *equivalence relation* is a relation that is reflexive, symmetric and transitive.

- Some examples:
  - $x = y$
  - $x \equiv_k y$
  - $x$ has the same color as $y$
  - $x$ has the same shape as $y$. 
Equivalence Relations

• Most modern programming languages include some sort of hash table data structure.
  • Java: HashMap
  • C++: std::unordered_map
  • Python: dict
• If you insert a key/value pair and then try to look up a key, the implementation has to be able to tell whether two keys are equal.
• Although each language has a different mechanism for specifying this, many languages describe them in similar ways...
Equivalence Relations

“The equals method implements an equivalence relation on non-null object references:

- It is reflexive: for any non-null reference value \( x \), \( x.equals(x) \) should return true.
- It is symmetric: for any non-null reference values \( x \) and \( y \), \( x.equals(y) \) should return true if and only if \( y.equals(x) \) returns true.
- It is transitive: for any non-null reference values \( x \), \( y \), and \( z \), if \( x.equals(y) \) returns true and \( y.equals(z) \) returns true, then \( x.equals(z) \) should return true.”

Java 8 Documentation
Equivalence Relations

“The equals method implements an equivalence relation on non-null object references:

• It is reflexive: for any non-null reference value x, x.equals(x) should return true.

• It is symmetric: for any non-null reference values x and y, x.equals(y) should return true if and only if y.equals(x) returns true.

• It is transitive: for any non-null reference values x, y, and z, if x.equals(y) returns true and y.equals(z) returns true, then x.equals(z) should return true.”

Java 8 Documentation
Equivalence Relations

“Each unordered associative container is parameterized by Key, by a function object type Hash that meets the Hash requirements (17.6.3.4) and acts as a hash function for argument values of type Key, and by a binary predicate Pred that induces an equivalence relation on values of type Key. Additionally, unordered_map and unordered_multimap associate an arbitrary mapped type T with the Key.”

C++14 ISO Spec, §23.2.5/3
Equivalence Relations

“Each unordered associative container is parameterized by Key, by a function object type Hash that meets the Hash requirements (17.6.3.4) and acts as a hash function for argument values of type Key, and by a binary predicate Pred that induces an equivalence relation on values of type Key. Additionally, unordered_map and unordered_multimap associate an arbitrary mapped type T with the Key.”

C++14 ISO Spec, §23.2.5/3
Equivalence Relation Proofs

• Let's suppose you've found a binary relation $R$ over a set $A$ and want to prove that it's an equivalence relation.

• How exactly would you go about doing this?