Binary Relations
Part One
Outline for Today

• *Binary Relations*
  • Reasoning about connections between objects.

• *Equivalence Relations*
  • Reasoning about clusters.

• *A Fundamental Theorem*
  • How do we know we have the “right” definition for something?
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:
    \( p \) 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 “made of zeros and ones.”
What exactly is a binary relation?
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 \( aRb \).
  
  \[
  3 = 3 \quad 5 < 7 \quad \emptyset \subseteq \mathbb{N}
  \]
- If \( R \) is a binary relation over \( A \) and it does not hold for the pair \((a, b)\), we write \( aR\neg b \).
  
  \[
  4 \neq 3 \quad 4 \lesssim 3 \quad \mathbb{N} \nsubseteq \emptyset
  \]
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 $5 \not< 3$.
  • In some relations order is irrelevant; more on that later.

• Relations are always defined relative to some underlying set.
  • It's not meaningful to ask whether $\odot \subseteq 15$, for example, since $\subseteq$ is defined over sets, not arbitrary objects.
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: the relation \( a \mid b \) (meaning “\( a \) divides \( b \)”) 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: the relation $a \neq b$ 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: the relation $a = b$ 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: 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.
Below is a picture of a binary relation $R$ over the set $\{1, 2, ..., 8\}$. Which of the following is a correct definition of the relation $R$?

A. $xRy$ if $x = 3$ and $y = 5$
B. $xRy$ if $y = x + 2$
C. $yRx$ if $y = x + 2$
D. $R = +2$
E. None of these
F. More than one of these

Answer at PollEv.com/cs103 or text CS103 to 22333 once to join, then A, B, C, D, E, or F.
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), and next time we'll see another (prerequisites).

• 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.
Partitions and Clustering

- If you have a set of data, you can often learn something from the data by finding a “good” partition of that data and inspecting the partitions.
  - Usually, the term *clustering* is used in data analysis rather than *partitioning*.
- Interested to learn more? Take CS161 or CS246!
What's the connection between partitions and binary relations?
∀a ∈ A. aRa

∀a ∈ A. ∀b ∈ A. (aRb → bRa)

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

• Some relations always hold from any element 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:

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

  ("Every element is related to itself.")
∀a ∈ A. aRa
(“Every element is related to itself.”)
∀a ∈ A. aRa

("Every element is related to itself.")
\[ \forall a \in A. \ aRa \]

(“Every element is related to itself.”)

This means that \( R \) is not reflexive, since the first-order logic statement given below is not true.
Reflexivity is a property of relations, not individual objects.

\[ \forall a \in \text{??}. \quad a \sim a \]
Symmetry

- In some relations, the relative order of the objects doesn't matter.
- 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 $R$:

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

("If $a$ is related to $b$, then $b$ is related to $a$.")
∀a ∈ A. ∀b ∈ A. (aRb → bRa)
(“If a is related to b, then b is related to a.”)
Is This Relation Symmetric?

∀a ∈ A. ∀b ∈ A. (aRb → bRa)

("If a is related to b, then b is related to a.")
∀a ∈ A. ∀b ∈ A. (aRb → bRa)
("If a is related to b, then b is related to a.")
Transitivity

• Many relations can be chained together.
• 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 \textit{transitive}.
• A binary relation $R$ over a set $A$ is called \textit{transitive} if the following first-order statement is true about $R$:

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

("Whenever $a$ is related to $b$ and $b$ is related to $c$, we know $a$ is related to $c$.")
Transitivity Visualized

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

("Whenever a is related to b and b is related to c, we know a is related to c.")
Is This Relation Transitive?

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

(“Whenever a is related to b and b is related to c, we know a is related to c.”)
∀a ∈ A. ∀b ∈ A. ∀c ∈ A. (aRb ∧ bRc → aRc)
("Whenever a is related to b and b is related to c, we know a is related to c.")

Is this relation transitive?

Answer at PollEv.com/cs103 or text CS103 to 22333 once to join, then Y or N.
Is This Relation Transitive?

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

("Whenever a is related to b and b is related to c, we know a is related to c.")
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 $y$. 
Binary relations give us a common language to describe common structures.
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

“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
Time-Out for Announcements!
Interpreting your Pset 1 Grade

- 25%ile: 60/80 (75%)
- Median: 67/80 (82.7%)
- 75%ile: 74/80 (92.5%)
Research Info Session

- CURIS (Undergraduate Research Institute “in” CS—har har har har) is a summer research experience in our dept
- Unbelievable cutting-edge projects
- See if grad school might be of interest
- Learn more:

  Tuesday, 1/30 at 5:30pm in Gates 219
Back to CS103!
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?
An Example Relation

• Consider the binary relation ~ defined over the set \( \mathbb{Z} \):

\[
a \sim b \quad \text{if} \quad a + b \text{ is even}
\]

• Some examples:

\[
0 \sim 4 \quad 1 \sim 9 \quad 2 \sim 6 \quad 5 \sim 5
\]

• Turns out, this is an equivalence relation! Let's see how to prove it.

We can binary relations by giving a rule, like this:

\[
a \sim b \quad \text{if} \quad \text{some property of } a \text{ and } b \text{ holds}
\]

This is the general template for defining a relation. Although we're using “if” rather than “iff” here, the two above statements are definitionally equivalent. For a variety of reasons, definitions are often introduced with “if” rather than “iff.” Check the “Mathematical Vocabulary” handout for details.
What properties must ~ have to be an equivalence relation?

Reflexivity  
Symmetry  
Transitivity

Let's prove each property independently.
\( a \sim b \quad \text{if} \quad a + b \text{ is even} \)

**Lemma 1:** The binary relation \( \sim \) is reflexive.

**Proof:**

What is the formal definition of reflexivity?

\[ \forall a \in \mathbb{Z}. \ a \sim a \]

Therefore, we'll choose an arbitrary integer \( a \), then go prove that \( a \sim a \).
Lemma 1: The binary relation ~ is reflexive.

Proof: Consider an arbitrary \( a \in \mathbb{Z} \). We need to prove that \( a \sim a \). From the definition of the ~ relation, this means that we need to prove that \( a + a \) is even.

To see this, notice that \( a + a = 2a \), so the sum \( a + a \) can be written as \( 2k \) for some integer \( k \) (namely, \( a \)), so \( a + a \) is even. Therefore, \( a \sim a \) holds, as required. ■
Lemma 2: The binary relation \( \sim \) is symmetric.

\[
a \sim b \quad \text{if} \quad a + b \text{ is even}
\]

Which of the following works best as the opening of this proof?

A. Consider any integers \( a \) and \( b \). We will prove \( a \sim b \) and \( b \sim a \).
B. Pick \( \forall a \in \mathbb{Z} \) and \( \forall b \in \mathbb{Z} \). We will prove \( a \sim b \rightarrow b \sim a \).
C. Consider any integers \( a \) and \( b \) where \( a \sim b \) and \( b \sim a \).
D. Consider any integer \( a \) where \( a \sim a \).
E. The relation \( \sim \) is symmetric if for any \( a, b \in \mathbb{Z} \), we have \( a \sim b \rightarrow b \sim a \).
F. Consider any integers \( a \) and \( b \) where \( a \sim b \). We will prove \( b \sim a \).

Answer at PollEv.com/cs103 or text CS103 to 22333 once to join, then A, B, C, D, E, or F.
Lemma 2: The binary relation $\sim$ is symmetric.

Proof:

What is the formal definition of symmetry?

$$\forall a \in \mathbb{Z}. \forall b \in \mathbb{Z}. (a \sim b \rightarrow b \sim a)$$

Therefore, we'll choose arbitrary integers $a$ and $b$ where $a \sim b$, then prove that $b \sim a$. 
\[ a \sim b \quad \text{if} \quad a+b \text{ is even} \]

**Lemma 2:** The binary relation \( \sim \) is symmetric.

**Proof:** Consider any integers \( a \) and \( b \) where \( a \sim b \). We need to show that \( b \sim a \).

Since \( a \sim b \), we know that \( a+b \) is even. Because \( a+b = b+a \), this means that \( b+a \) is even. Since \( b+a \) is even, we know that \( b \sim a \), as required. ■
Lemma 3: The binary relation $\sim$ is transitive.

Proof:

What is the formal definition of transitivity?

$$\forall a \in \mathbb{Z}. \forall b \in \mathbb{Z}. \forall c \in \mathbb{Z}. \ (a \sim b \land b \sim c \rightarrow a \sim c)$$

Therefore, we'll choose arbitrary integers $a$, $b$, and $c$ where $a \sim b$ and $b \sim c$, then prove that $a \sim c$. 
Lemma 3: The binary relation ~ is transitive.

Proof: Consider arbitrary integers $a$, $b$ and $c$ where $a \sim b$ and $b \sim c$. We need to prove that $a \sim c$, meaning that we need to show that $a+c$ is even.

Since $a \sim b$ and $b \sim c$, we know that $a+b$ and $b+c$ are even. This means there are integers $k$ and $m$ where $a+b = 2k$ and $b+c = 2m$. Notice that

$$(a+b) + (b+c) = 2k + 2m.$$ 

Rearranging, we see that

$$a+c + 2b = 2k + 2m,$$

so

$$a+c = 2k + 2m - 2b = 2(k+m-b).$$

So there is an integer $r$, namely $k+m-b$, such that $a+c = 2r$. Thus $a+c$ is even, so $a \sim c$, as required. ■
\[ a \sim b \quad \text{if} \quad a+b \text{ is even} \]

**Lemma 1:** The binary relation \( \sim \) is reflexive.

**Proof:** Consider an arbitrary \( a \in \mathbb{Z} \). We need to prove that \( a \sim a \). From the definition of the \( \sim \) relation, this means that we need to prove that \( a+a \) is even.

To see this, notice that \( a+a = 2a \), so the sum \( a+a \) can be written as \( 2k \) for some integer \( k \) (namely, \( a \)), so \( a+a \) is even. Therefore, \( a \sim a \) holds, as required. ■

The formal definition of reflexivity is given in first-order logic, but this proof does not contain any first-order logic symbols!
\[ a \sim b \quad \text{if} \quad a+b \text{ is even} \]

**Lemma 2:** The binary relation \( \sim \) is symmetric.

**Proof:** Consider any integers \( a \) and \( b \) where \( a \sim b \). We need to show that \( b \sim a \).

Since \( a \sim b \), we know that \( a+b \) is even. Because \( a+b = b+a \), this means that \( b+a \) is even. Since \( b+a \) is even, we know that \( b \sim a \), as required. ■

The formal definition of symmetry is given in first-order logic, but this proof does not contain any first-order logic symbols!
\[ a \sim b \quad \text{if} \quad a+b \text{ is even} \]

**Lemma 3:** The binary relation \( \sim \) is transitive.

**Proof:** Consider arbitrary integers \( a, b \) and \( c \) where \( a \sim b \) and \( b \sim c \). We need to prove that \( a \sim c \), meaning that we need to show that \( a+c \) is even.

Since \( a \sim b \) and \( b \sim c \), we know that \( a+b \) and \( b+c \) are even. This means there are integers \( k \) and \( m \) where \( a+b = 2k \) and \( b+c = 2m \). Notice that

\[
(a+b) + (b+c) = 2k + 2m.
\]

Rearranging, we see that

\[
a+c + 2b = 2k + 2m,
\]

so

\[
a+c = 2k + 2m - 2b = 2k + 2m - 2(b-k) = 2r.
\]

So there is an integer \( r \), namely \( k + m - b \), such that \( a+c = 2r \). Thus \( a+c \) is even.

The formal definition of transitivity is given in first-order logic, but this proof does not contain any first-order logic symbols!
First-Order Logic and Proofs

- First-order logic is an excellent tool for giving formal definitions to key terms.
- While first-order logic *guides* the structure of proofs, it is *exceedingly rare* to see first-order logic in written proofs.
- Follow the example of these proofs:
  - Use the FOL definitions to determine what to assume and what to prove.
  - Write the proof in plain English using the conventions we set up in the first week of the class.
- *Please, please, please, please, please, please, please internalize the contents of this slide!*