Discrete Mathematics

Topic 04 — Relations and Functions

Lecture 02 — Relations on a Set

Dr Kieran Murphy

Department of Computing and Mathematics,
Waterford IT.
(kmurphy@wit.ie)

Autumn Semester, 2021

Outline

- Properties of relations on sets

Outline

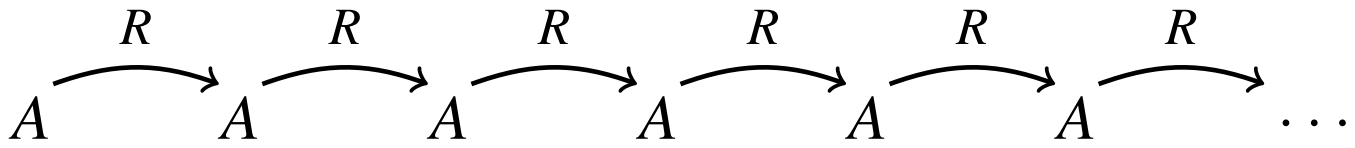
Relations on a Set

We now will focus on relations where the source and target sets are equal. Recall our definition . . .

Definition 1 (Relation on a Set)

A relation from set A to set A is called a relation on A

- Such relations are important because they occur frequently in practice
  - In cryptography we have relations from $\mathbb{N}$ to $\mathbb{N}$.
  - In 3D graphics/animation we are have relations from $\mathbb{R}^3$ to $\mathbb{R}^3$, etc.
- Because the target is equal to the source, we can reapply the same relation, taking the output of one application as the input to the next application.



- This feature gives rise to powerful algorithms: fixed point methods, hashing, block chaining, etc.

Reflexive Property

Definition 2 (Reflexive)

Let R be a relation on set A. Then R is said to be reflexive iff

$$(a,a) \in R \qquad \forall a \in A$$

On the set of positive integers, $A = \{1, 2, \ldots\}$ the following relations are reflexive

- $R =$"divides", i.e., $\{(a_1, a_2) \in R$ iff $a_1 \mid a_2$ where $a_1, a_2 \in A\}$
  (R is reflexive since every positive integer divides itself)
- $R =$ "less or equal to", i.e., $\{(a_1, a_2) \in R$ iff $a_1 \leq a_2$ where $a_1, a_2 \in A\}$
  (R is reflexive since $a \leq a$ for every positive integer.)

and the following relations are not reflexive

- $R =$ "less than" $(1,5) \in R$ but $(5,1) \notin R$
- $R =$ "is half" $(3,6) \in R$ but $(6,3) \notin R$

Symmetric Property

Definition 3 (Symmetric/Anti-symmetric)

Let R be a relation on set A. Then R is said to be symmetric iff

$$(a_1, a_2) \in R \Rightarrow (a_2, a_1) \in R \qquad \forall a_1, a_2 \in A$$

And R is said to be anti-symmetric iff

$$(a_1, a_2) \in R \ \wedge \ (a_2, a_1) \in R \quad \Rightarrow \quad a_1 = a_2 \qquad \forall a_1, a_2 \in A$$

- Note that a relation could be both/either one/neither of symmetric or anti-symmetric.
- A symmetric relation is one in which if a_1 is related to a_2, then a_2 *must* be related to a_1.
- The standard approach to proving a relation is anti-symmetric is via proof by contradiction:
  - Start with assuming $(a_1, a_2) \in R$ and $(a_2, a_1) \in R$ with $a_1 \neq a_2$
  - Prove that $a_1 \neq a_2$ leads to a contradiction.

Example 4

Example 4

Consider the relation $R = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$. Is R reflexive? symmetric? anti-symmetric?

(Reflexive) R is not reflexive since, for example $(2, 2) \notin R$.

(Symmetric) R is symmetric since $x^2 + y^2 = y^2 + x^2$. Hence, if $(x, y) \in R$ then so is $(y, x) \in R$.

(Anti-symmetric) R is not anti-symmetric since $(0, 1) \in R$ and $(1, 0) \in R$ but $1 \neq 0$.

Transitive Property

Definition 5 (Transitive)

Let R be a relation on set A. Then R is said to be transitive iff

$$(a_1, a_2) \in R \ \wedge \ (a_2, a_3) \in R \quad \Rightarrow \quad (a_1, a_3) \in R$$

- Relation like “less than”, “greater than” , “equal to”, “ancestor of ”, etc. are transitive.

Equivalence Relation

Definition 6 (Equivalence Relation/Equivalence Classes)

Let R be a relation on a set A. Then we say R is an equivalence relation if R is

- Reflexive
- Symmetric
- Transitive

Then R "divides" the set A into equivalence classes, $A_1, A_2, \ldots$, where two elements, a_1 and a_2, are in the same class iff $(a_1, a_2) \in R$. And

$$A_i \cap A_j = \emptyset \quad \text{when } i \neq j$$

and

$$A_1 \cup A_2 \cup A_3 \cdots = A$$

i.e, the sets $A_1, A_2, \ldots$ are disjoint and exhaustive on A.

Example 7

Example 7

Let A be the set of all people in Ireland. Consider the relation, $(x, y) \in R$ if x and y have the same surname (that is, last name). Then R is an equivalence relation:

(Reflexive) R is reflexive since any person, x, has the same surname as his/her self.

(Symmetric) R is symmetric since if x has the same surname as y then y has the same surname as x.

(Transitive) R is transitive since if x has the same surname as y and y has the same surname as z then x has the same surname as z.

Thus R is an equivalence relation.

- The equivalence classes are all those people with surname “Applby”, all those people with surname “Murphy”, etc.

Example 8

Example 8

Let A be the set of all people in Ireland. Consider the relation, $(x, y) \in R$ if x and y have at least one biological parent in common.
It is easy to see that this relation is reflexive and symmetric:

(Reflexive) R is reflexive since any person, x, has at least least one biological parent in common with him/her self.

(Symmetric) R is symmetric since if x has at least least one biological parent in common with y, then y has at least least one biological parent in common with x.

However, the relation is not transitive:

(~~Transitive~~) R is not transitive. For example, say Jack and Jill had a child called Larry, then Jack and Bo Beep had a child called Curly, and finally Bo Beep and Jack Sprat had a child called Mo. Then Larry and Curly are related, and Curly and Mo are related by Larry and Mo are not related.

Other Properties

Finally the following two properties are give for completeness

Definition 9 (Irreflective/Asymmetric)

- A relation on set A is irreflective if

$$(a, a) \notin R \qquad \forall a \in A$$

i.e., For all a in A, a is not related to itself.

- A relation on set A is asymmetric if

$$(a_1, a_2) \in R \Rightarrow (a_2, a_1) \notin R \qquad \forall a_1, a_2 \in A,$$

i.e., For all a_1 and a_2 in A, if a_1 is related to a_2, then a_2 is not related to a_1,

Lemma 10

A relation R on set A is asymmetric if and only if

- *R is irreflexive*
- *R is antisymmetric*

Graphical Representation of Relations using Digraphs

We will now cover an alternative graphical representation of relations that is useful when studying reflexive, symmetry and transitive properties. This representation is based on directed graphs, called a digraph*.

Digraph representation of a Relation

Consider a relation R on the set A:

- Each element in $a \in A$ is drawn as a point/circle called a vertex/node/point.
- If $(a_1, a_2) \in R$ then a (curved) line with an arrow is drawn from a_1 to a_2, these are called edges/arcs/lines.
- The position of the vertex and the layout of the edges is not important, only which vertices have directed edges towards which vertices.

*We will look at the area of graphs in more detail in weeks 9-10.

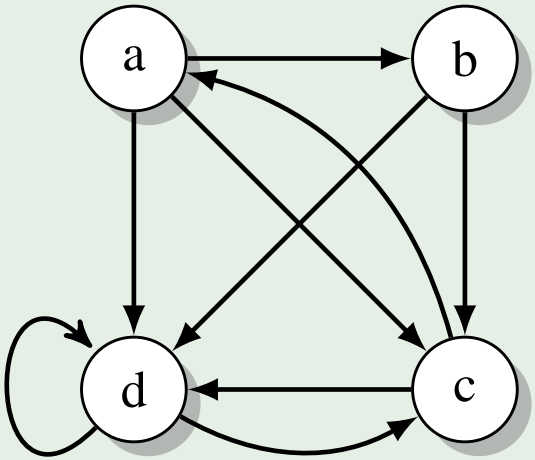
Example 11

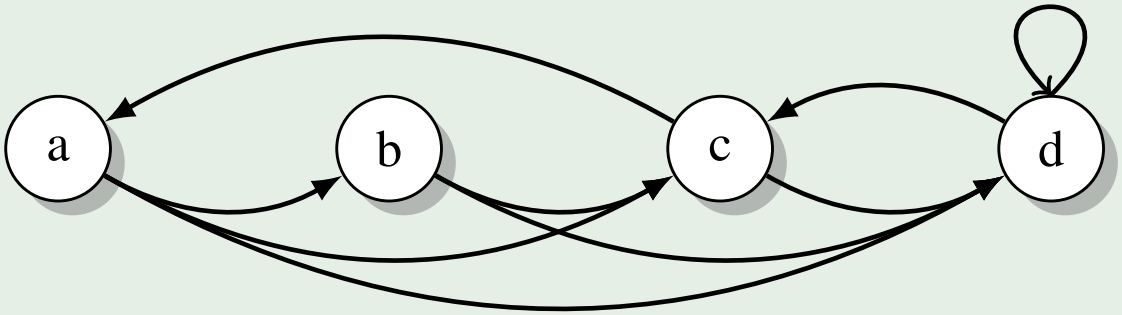
Example 11

Let $A = \{a, b, c, d\}$ and let R be a relation on A defined as

$$R = \{(a,b), (a,c), (a,d), (b,c), (b,d), (c,a), (c,d), (d,c), (d,d)\}$$

Possible digraph representations include





Digraphs and Properties of Relations I

Reflexivity

In a digraph, a relation is reflexive if and only if every vertex has a self loop[†].

Symmetry

In a digraph, a relation is symmetric if and only if for every edge from x to y there is also a corresponding edge from y to x. (i.e., bidirectional edge pairs)

Anti-symmetry

In a digraph, a relation is anti-symmetric if and only if, for every edge from x to y there is never a corresponding edge from y to x, whenever x and y are distinct vertices (i.e., we ignore self loops).

Transitivity

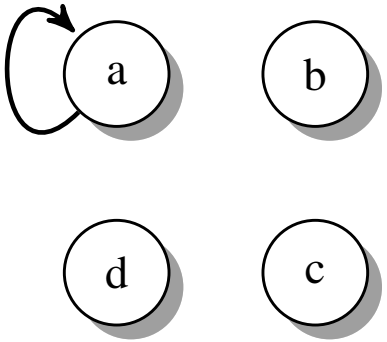
In a digraph, a relation is transitive if for every pair of edges (x, y) and (y, z) there is also a edge (x, z).

[†] A self loop is an edge from a vertex to itself.

Digraphs and Properties of Relations II

Consider the following relations on $A = \{a, b, c, d\}$. Determine their properties

$R = \{(a, a)\}$



reflexive ✘

symmetric ✔

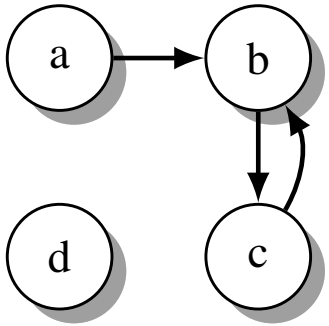
anti-symmetric ✔

transitive ✔

irreflexive ✘

asymmetric ✘

$R = \{(a, b), (b, c), (c, b)\}$



reflexive ✘

symmetric ✘

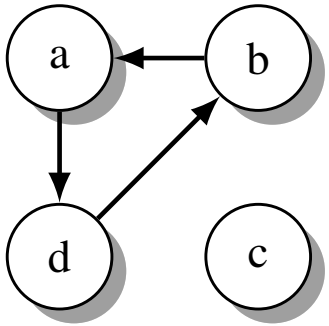
anti-symmetric ✘

transitive ✘

irreflexive ✔

asymmetric ✘

$R = \{(a, d), (d, b), (b, a)\}$



reflexive ✘

symmetric ✘

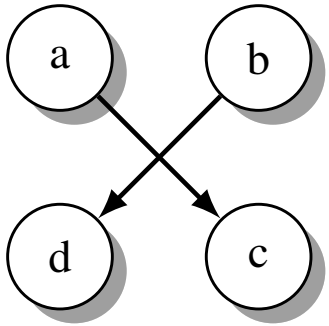
anti-symmetric ✔

transitive ✘

irreflexive ✔

asymmetric ✔

$R = \{(a, c), (b, d)\}$



reflexive ✘

symmetric ✘

anti-symmetric ✔

transitive ✔

irreflexive ✔

asymmetric ✔

Closure of Relations

Given a relation that does not have a required property we often want to determine the minimal changes (additions) needed so that the modified relation does have the required property.

Definition 12 (Closure of Relations)

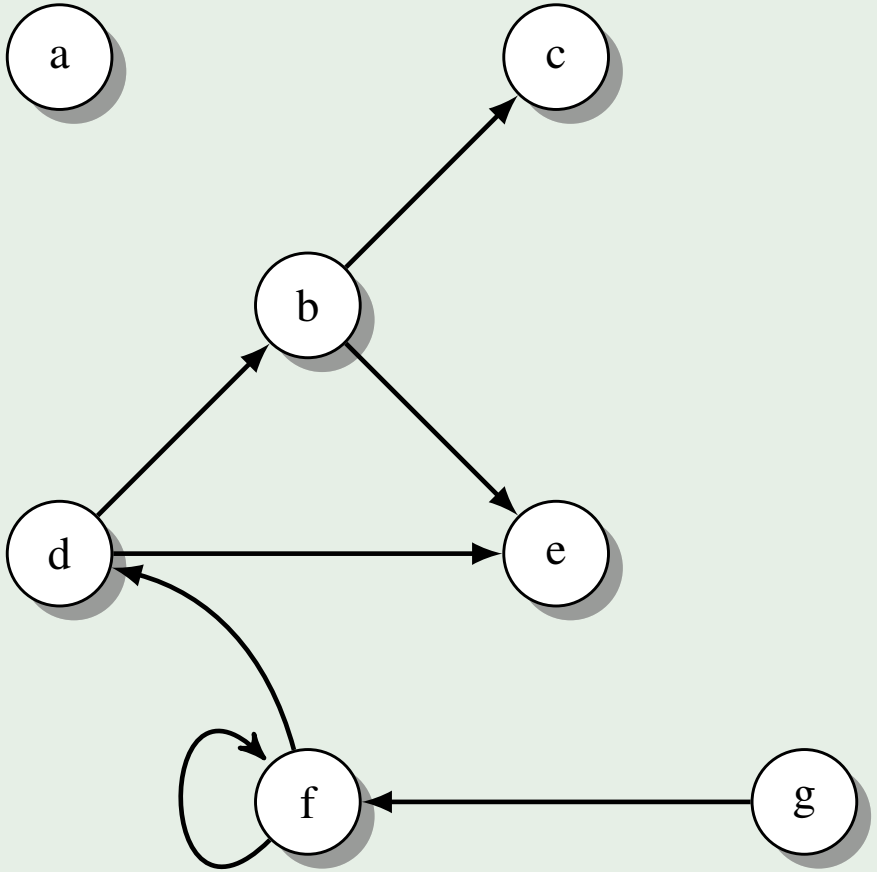
For any property X, the "X closure" of relation R is defined as the "smallest" superset of R that has the given property.

- The reflexive closure of a relation R on A, is obtained by adding (a, a) to R for each $a \in A$.
- The symmetric closure of a relation R on A, is obtained by adding (a_2, a_1) to R for each (a_1, a_2) in R.
- The transitive closure of a relation R on A, is obtained by repeatedly adding (a_1, a_3) to R for each (a_1, a_2) and (a_2, a_3) in R.

Example 13

Example 13

Find the symmetric closure, reflective closure and transitive closure of the relation represented in the following digraph.



Equivalence Classes

Earlier we discussed that if a relation, R, is reflexive, symmetric and transitive on set A then R "divides" the set A into equivalence classes, $A_1, A_2, \ldots$, where two elements, a_1 and a_2, are in the same equivalence class iff $(a_1, a_2) \in R$.

The equivalence classes are disjoint and exhaustive on A, .i.e.,

- (disjoint) $A_i \cap A_j = \emptyset$ when $i \neq j$
- (exhaustive on A) $A_1 \cup A_2 \cup A_3 \cdots = A$

Examples

- The "is equal to" relation over integers is an equivalence relation. The resulting equivalence classes (all singletons) are the individual integers.
- The "have same length" over strings is an equivalence relation. The resulting equivalence classes are strings of length zero, length one, etc.
- The "same remainder when divided by two" over integers is an equivalence relation. The resulting equivalence classes are the odd and even integers.

Iterating Relations

We often want to apply a relation a fixed number of times (for loops) or until some condition is satisfied (while loops). To represent this in Mathematics we define the n^{th} power of a relation

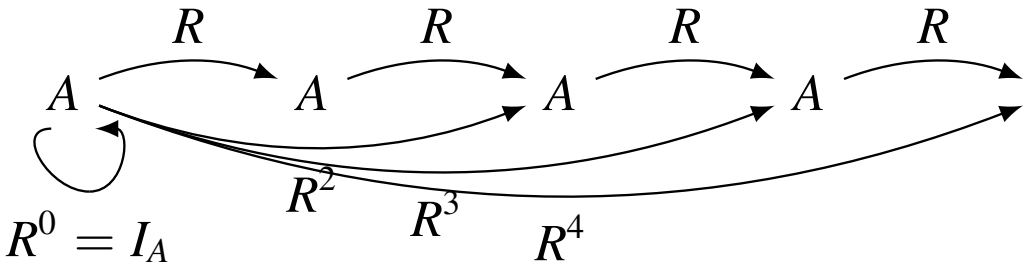
Definition 14 (Power of a Relation)

The n^{th} power of a relation R on a set A is defined recursively as

$$R^n = \begin{cases} I_A & n = 0, \\ R^{n-1} \circ R & \text{for all } n > 1 \end{cases}$$

where

- I_A is the identity relation on A, i.e, $I_A = \{(a, a) \mid a \in A\}$
- The "$\circ$" means apply after (so apply from right to left).

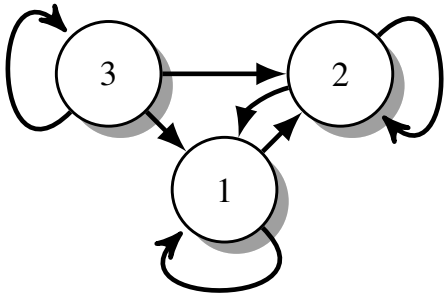


Review Exercises 1 (Properties of Relations on a Set)

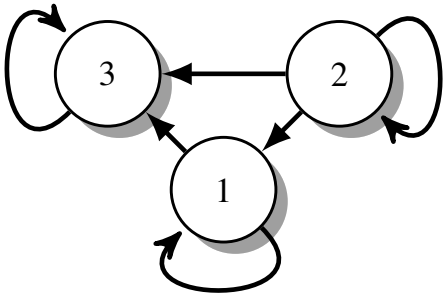
Question 1:
Consider the relations represented in the following graphs.

- Determine whether the given relations are reflexive, symmetric, antisymmetric, or transitive.
- Determine which relations are asymmetric, irreflexive.
- Which of the graphs are of equivalence relations?
- Construct the transitive closure of each relation.

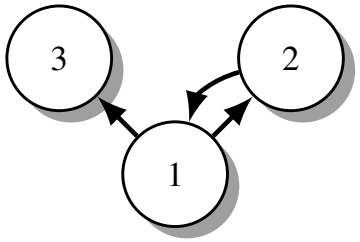
(a)



(b)



(c)



Question 2:
Consider the relation on $\{1,2,3,4,5,6\}$ defined by $R = \{(i,j) :| i-j |= 2\}$.You

a) Is R reflexive?

b) Is R symmetric?

c) Is R transitive?

d) Draw a digraph of R.

Outline

A Sample Application — Specification

We have covered lots[‡] concepts and definitions relating to sets and relations. It might help your appreciation of these abstract ideas if you see then being applied to a realistic problem.

Problem 15

Write a program to randomly generate English like text using arbitrary valid English text as a starting point.

Development

- Get sample text to use — *Land of Stories* by Chris Colfer.
- Read text in and strip punctation.
- Build set of all words in document. (this is my set A)
- Build relation between pairs of words based on relation "follows".
- Start generating new text by
  - Pick a random word
  - Repeatedly apply relation "follows" by selecting a word that follows the current word.

[‡] probably too many

A Sample Application — Sample Run

Original

"Once upon a time" Mrs. Peters said to her sixth-grade class. "These are the most magical words our world has ever known and the gateway into the greatest stories ever told. They're an immediate calling to anyone who hears them—a calling into a world where everyone is welcome and anything can happen. Mice can become men, maids can become princesses, and they can teach valuable lessons in the process."
Alex Bailey eagerly sat straight up in her seat. She usually enjoyed her teacher's lessons, but this was something especially close to her heart. "Fairy tales are much more than silly bedtime stories," the teacher continued. "The solution to almost every problem imaginable can be found in the outcome of a fairy tale. Fairy tales are life lessons disguised with colorful characters and situations.
...

Automaticaly Generated Text

looking for our own happily-ever-afters. he did. said with a student raised her hand. If she and short strawberry-blonde hair and sat straight up in need. The solution to say or do. She didn't matter how many kidnappings if everyone in school...mostly trouble staying awake. Mrs. Peters went to the room while his pack made himself from all missing the maiden because of her hand. And why we?re having a lot. Mrs. Peters rolled her eyes were all the wolf was a sister by any child when you know enough about your mother, do you could follow her teacher's lessons, but unlike his brain tried her real Little Red Riding Hood became queen, Mrs. Peters said. Once they would be much as a nap. Alex explained. the wolf was the answer Mrs. Peters said, as comfortable as her brother. ...

A Sample Application — Implementation I

garbage_generator.py

```
3  # The random module will be used to generate the output text
4  import random
5
6  # Read in original text and convert line breaks to space
7  data = open("land_of_stories.txt","r").read()
8  data = data.replace("\n", "␣").replace("\"", "␣")
9
10 # To store the relation "follows"
11 # NOTE: This is a dictionary not a set (but ignore that for now)
12 words = {}
13
14 # Split document into words and store the word pairs based on
15 # the "follows" relation into words
16 previous = None
17 for word in data.split("␣"):
18     if word not in words: words[word] = []
19     if previous is not None:
20         words[previous].append(word)
21     previous = word
```

A Sample Application — Implementation II

garbage_generator.py

```
23 # To stote the generated text
24 content = ""
25
26 # Set of word in document
27 # converted set to list (order matters) to pick at random (line 35)
28 start = list(words.keys())
29
30 # While the current word has word that "follow" in the original text
31 #     Pick one of "following" words and append this to output
32 #     Replace current word by new word
33 next_word_options = start
34 while len(next_word_options)>0:
35     word = random.choice(next_word_options)
36     content += word + "␣"
37     next_word_options = words[word]
38
39 # Finally output generated garbage
40 print (content)
```