Chapters 1, 2, 10, 3

[1] Speaking Mathematically

▼ [1.1] Variables

- **Variables** are a mathematical placeholder meant to represent an unknown value or to use to reference numerous elements of a single set

$$\text{Is there a number } x \text{ with the property: } 2x+3=x^2?$$

$$\begin{gathered}\text{Are there numbers with the property that the sum}\\ \text{of their squares equals the square of their sum?}\\ \downarrow\\ \text{Are there some numbers } a \text{ and } b \text{ with the property that } a^2+b^2=(a+b)^2\end{gathered}$$

- **Variables** in computer science are references to the memory, allowing for important values to remain reference-able for later or be changed

```
/*
In Java for instance, a keyword is used to
define the variable type before it is named and established as a ref
erence
*/
int num;
boolean isOpen;
String str;
```

Mathematical Statements

- **Universal statements** state that a certain property is true for all elements in a set
  - *"All positive numbers are greater than 0."*
- **Conditional statement** state that one thing must be true if another particular thing is
  - *"If 378 is divisible by 18, then 378 is divisible by 6."*

- **Existential statements** state that there is at least one thing for which the property is true
    - *"There is a prime number that is even."*

Universal Conditional Statements

- Usually contains the phrase: "for every"
- **Universal conditional statements** are statements that is both universal and conditional
    - *"For every animal a, if a is a dog, then a is a mammal."*
    - They can be written in ways that make them appear purely universal or purely conditional.
        - Explicit conditional and implicit universal
            - *"If a is a dog, then a is a mammal."*
            - *"If an animal is a dog, then the animal is a mammal."*
        - Implicit conditional and explicit universal
            - *"For every dog a, a is a mammal."*
            - *"All dogs are mammals."*

Universal Existential Statements

- **Universal existential statements** are statements where the first part is universal and the second part is existential
    - *"Every real number has an additive inverse."*
    - *"For every real number r, there is an additive inverse for r."*
    - *"For every real number r, there is a real number s such that s is an additive inverse for r."*

Existential Universal Statements

- **Existential universal statements** are statements where the first part is existential and the second part is universal
    - *"There is a positive integer that is less than or equal to every positive integer."*
    - *"There is a positive integer m that is less than or equal to every positive integer."*
    - *"There is a positive integer m such that every positive integer is greater than or equal to m."*

    - *"There is a positive integer m with the property that for every positive integer n, $m \leq n$."*

- Many important mathematical concepts require the uses of the phrases: "for every," "there is," and "if-then"
    - For example, if $a_1,\ a_2,\ a_3, \ldots$ is a sequence of real numbers, saying that $\lim_{n\to\infty} a_n = L$ means that:
        - **For every** positive real number ϵ, **there is** an integer N such that **for every** integer n, **if** $n > N$ **then** $-\epsilon < a_n - L < \epsilon$

▼ [1.2] The Language of Sets

$$\begin{gathered}\text{Small Set}\\ \{\,1,2,3\,\}\\ \\ \text{Bigger Set (Up to 100)}\\ \{\,1,2,3,\ldots,100\,\}\\ \\ \text{Infinite Set (All positive integers)}\\ \{\,1,2,3,\ldots\,\}\end{gathered}$$

- The **axiom of extension** states that a set is completely determined by what their elements are and not their order or potential duplication of elements

Set-Roster Notation

$$\begin{gathered}\text{Sets containing } 1,2,3\\ \{\,1,2,3\,\}\\ \{\,3,2,1\,\}\\ \{\,1,1,2,3,3,3\,\}\end{gathered}$$

- The 3 sets above are the same set as defined by the aforementioned axiom of extension
- **Cardinality** is the number of unique elements within a set
- $\{0\} \neq 0$ as one is a set comprised of the element 0 while the other is just the symbol representing 0
- Some common number sets are represented as singular letters with the blackboard bold font

$$\text{Set of all real numbers } \mathbb{R}$$
$$\text{Set of all rational numbers } \mathbb{Q}$$
$$\text{Set of all integers } \mathbb{Z}$$

- Superscripts ($\mathbb{R}^+,\ \mathbb{R}^-,\ \mathbb{R}^{\text{nonneg}}$) indicate that all elements of the set are positive, negative, or nonnegative, respectively
    - nonneg is used in this course, but the most popular notation is $\mathbb{R}_{\geq 0}$
- Sometimes, $\mathbb{Z}_{\geq 0}$ is denoted as $\mathbb{N}$, the set of all natural numbers
    - In other conventions, $\mathbb{Z}^+ = \mathbb{N}$

Set-Builder Notation

$$\text{The set of all elements } x \text{ such that } P(x) \text{ is true}$$
$$\{\, x \in S \mid P(x) \,\}$$

Subsets

- The relation between sets is that of a **subset**. A set may be a subset of another if all of its elements are in the other set
    - For every element x, if $x \in A$ then $x \in B$ is denoted as

$$A \subseteq B$$

- **Proper subsets** only contain values of another set, but cannot have the exact same contents as the other set
    - A set is a subset of itself because they contain the same elements but cannot be a proper subset of itself and is denoted as
    - For every element x, if $x \in A$ then $x \in B$ and if $x \in B$ then $x \in A$ is denoted as

$$A \subset B$$

- To say that set A is not a subset of set B means that there is at least one element x such that $x \in A$ and $x \subsetneq B$ or

$$A \subsetneq B$$

- Here is an example below to illustrate subset relationships

$$A = \mathbb{Z}^+$$
$$B = \{\, n \in \mathbb{Z} \mid 0 \le n \le 100 \,\}$$
$$C = \{\, 100, 200, 300, 400, 500 \,\}$$

- $B \subseteq A$ is NOT a true statement because B includes 0, a nonnegative number while A is a set of all positive integers which does not include 0
- $C \subset A$ because all of its elements are within A but it lacks the positive integers in between the hundreds
- $C \subseteq B$ is NOT a true statement because B only goes up to 100 while C has 4 elements greater than 100
- $C \subseteq C$ because a set will share the same elements as itself

- There is a distinct difference for a set being *in* a set and a set being a *subset* of another set
  - A set being a subset of another refers to its elements being in the other set

$$\{\,1\,\} \subseteq \{\,1, 2\,\} \quad \text{but} \quad \{\,1\,\} \nsubseteq \{\,\{\,1\,\}, 2\,\}$$

  - A set being in another means that the set itself is an actual element of the other set

$$\{\,1\,\} \in \{\,\{\,1\,\}, 2\,\} \quad \text{but} \quad \{\,1\,\} \notin \{\,1, 2\,\}$$

Cartesian Product Notation

- Given elements a and b, the symbol (a, b) denotes the **ordered pair** consisting of a and b together with the specification that a is the first element of the pair and that b is the second element. Two ordered pairs (a, b) and (c, d) are equal if, and only if, a=c and b=d. Symbolically: $(a, b) = (c, d)$ means that $a = c$ and $b = d$

Cartesian Products

- An ordered n-tuple generalizes the notation for a pair to a set with a finite number of elements, taking both order and multiplicity into account
  - Ordered pairs consist of two elements while ordered triples consist of three elements

$$(x_1, x_2, \ldots, x_n)$$

- Two of these ordered n-tuples $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ may only be equal if $x_n = y_n$ with the aforementioned order and multiplicity mattering

- **Cartesian products** are the sets of all ordered n-tuples from multiplying two sets together
  - Given sets $A_1, A_2, \ldots, A_n$, the cartesian product is given as

$$A_1 \times A_2 \times A_3 = \{\, (a_1, a_2, \ldots, a_n) \mid a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n \,\}$$

  - Where a_n is some element in A_n
- Thus, if $A = \{\, x, y \,\}$, $B = \{\, 1, 2, 3 \,\}$, and $C = \{\, a, b \,\}$ then

$$\begin{aligned} A \times B &= \{\, (x,1), (y,1), (x,2), (y,2), (x,3), (y,3) \,\} \\ B \times A &= \{\, (1,x), (1,y), (2,x), (2,y), (3,x), (3,y) \,\} \\ A \times A &= \{\, (x,x), (y,x), (x,y), (y,y) \,\} \end{aligned}$$

$$\begin{aligned} A \times B \times C = \{\, & (x,1,a), (x,2,a), (x,3,a), \\ & (y,1,a), (y,2,a), (y,3,a), \\ & (x,1,b), (x,2,b), (x,3,b), \\ & (y,1,b), (y,2,b), (y,3,b) \,\} \end{aligned}$$

  - Cartesian products may be between cartesian products

$$\begin{aligned} (A \times B) \times C = \{ & \big((x,1),a\big), \big((x,2),a\big), \big((x,3),a\big), \\ & \big((y,1),a\big), \big((y,2),a\big), \big((y,3),a\big), \\ & \big((x,1),b\big), \big((x,2),b\big), \big((x,3),b\big), \\ & \big((y,1),b\big), \big((y,2),b\big), \big((y,3),b\big) \} \end{aligned}$$

- The number of elements in the cartesian product can be found by multiplying the number of elements in each set together
- **Strings** are ordered n-tuples that can be "wrapped" over a finite set with its characters consisting of the elements of the set it wraps
  - Thus, the number of distinct strings of length l over a finite set of size n is given by

$$n^l$$

  - For instance, given a string of length 3

$$A = \{ a, b \}$$

$$\begin{array}{ll} & \text{Strings} \\ aaa & (1) \\ aab & (2) \\ aba & (3) \\ baa & (4) \\ abb & (5) \\ bab & (6) \\ bba & (7) \\ bbb & (8) \end{array}$$

- **Null strings** have no characters and thus has a length of 0 and are denoted by

$$\lambda$$

- **Bit strings** wrap over the set

$$A = \{ 0, 1 \}$$

▼ [1.3] The Language of Relations and Functions

- Let A and B be sets
- A relation R from A to B is a subset of $A \times B$
- Given an ordered pair (x, y) in $A \times B$, x is related to R, written $x\ R\ y$, if and only if, (x, y) is in R
- The set A is called the **domain** of R and the set of B is called its **co-domain**

Relation Example

Let $A = \{ 1,\ 2 \}$ and $B = \{ 1,\ 2,\ 3 \}$, and define a relation R from A to B as follows:

$$\text{Given any } (x, y) \in A \times B,$$
$$(x, y) \in R \text{ means that } \frac{x-y}{2} \text{ is an integer.}$$

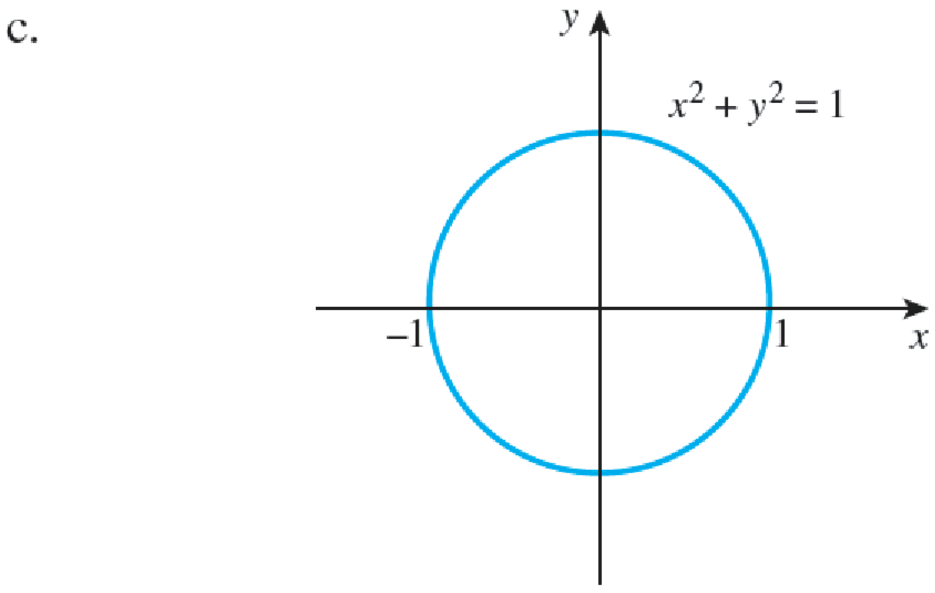
- $A \times B = \{ (1,1),\ (1,2),\ (1,3),\ (2,1),\ (2,2),\ (2,3) \}$, thus $R = \{ (1,1),\ (1,3),\ (2,2) \}$
- $1\ R\ 3$ is true because $(1,3) \in R$
- $2\ R\ 3$ is false because $(2,3) \not\in R$

- $2\ R\ 2$ is true because $(2,2) \in R$
- The domain is $\{1,\ 2\}$ (from A) and the co-domain is $\{1,\ 2,\ 3\}$ (from B)

Circle Relation

$$\text{Define a relation } C \text{ from } \mathbb{R} \text{ to } \mathbb{R} \text{ as follows:}$$
$$\text{for any } (x,y) \in R \times R,$$
$$(x,y) \in C \text{ means that } x^2 + y^2 = 1$$

- $(1,0) \in C$ is true because $1^2 + 0^2 = 1$
- $(0,0) \in C$ is false because $0^2 + 0^2 \neq 0$
- $\left(-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) \in C$ is true because $\left(-\frac{1}{2}\right)^2 + \left(-\frac{\sqrt{3}}{2}\right)^2 = \frac{1}{4} + \frac{3}{4} = 1$
- $-2\ C\ 0$ is false because $(-2,0) \notin C$
- $0\ C\ (-1)$ is true because $(0,-1) \in C$
- The domain and co-domain are both $\mathbb{R}$
- On a cartesian plane, C is graphed as:



Arrow Diagram of Relation

- Suppose R is a relation from a set A to a set B. The **arrow diagram** for R is obtained as follows:

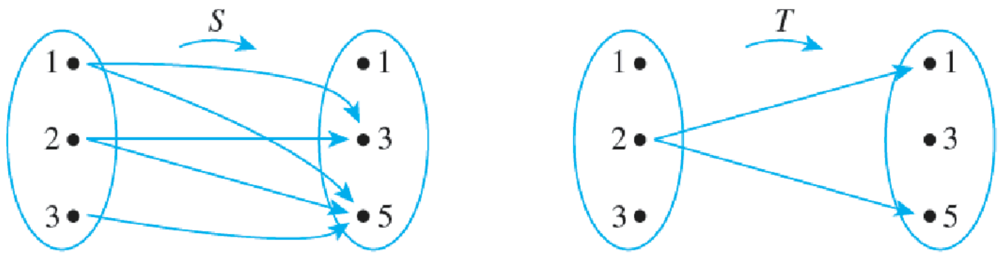
1. Represent the elements of A as points in one region and the elements of B as points in another region
2. For each x in A and each y in B, draw an arrow from x to y if, and only if, x is related to y by R

- Symbolically: draw an arrow from x to y if, and only if, and only if, $x \; R \; y$ and $(x, y) \in R$

$$\text{Let } A = \{1,\ 2,\ 3\} \text{ and } B = \{1,\ 2,\ 3\} \text{ and}$$
$$\text{define relations } S \text{ and } T \text{ from } A \text{ to } B \text{ as follows:}$$

$$(x, y) \in A \times B,$$
$$\text{for every } (x, y) \in S \text{ means that } x < y$$
$$T = \{(2, 1),\ (2, 5)\}$$

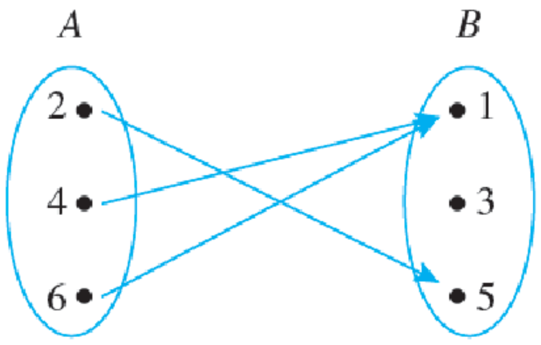
- This results in the following arrow diagram:



Functions

- A **function** F from a set A to a set B is a relation with domain A and co-domain B that satisfies the two following properties:
  1. For every element x in A, there is an element y in B such that $(x, y) \in F$
     a. Every element of A is the first element of an ordered pair of F
  2. For all elements x in A and y and z in B, if $(x, y) \in F$ and $(x, z) \in F$, then $y = z$
     a. No two distinct element pairs in F have the same first element
- The notation is as follows:
  - If A and B are sets and F is a function from A to B, then given any element x in A, the unique element in B that is related to x by F is denoted $F(x)$

- An example of a function is $T = \{\,(2,5),\ (4,1),\ (6,1)\,\}$ as demonstrated by this arrow diagram



Functions and Relations on Sets of Strings

$$\text{Let } A = \{\,a,\ b\,\} \text{ and let } S \text{ be the set of all strings over } A$$

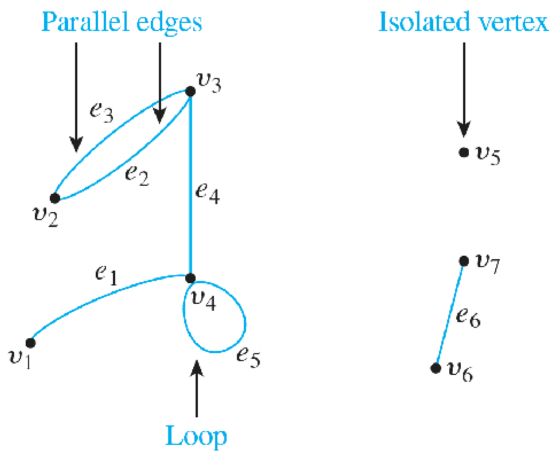
- Define a relation L from S to $\mathbb{Z}^{\text{nonneg}}$ as follows: for every string s in S and for every nonnegative integer n
  - $(s,n) \in L$ means that the length of s is n
  - L is a function because every string in S has only one length
  - $L(abaaba) = 6$
  - $L(bbb) = 3$
- Define a relation C from S to S as follows: for all strings s and t in S
  - $(s,t) \in C$ means that $t = as$ where as is the string obtained by appending a on the left of the characters in s
  - C is called **concatenation** by a on the left
  - C is a function because every string in S are only a's and b's and that adding an a on the left creates a new string that is also just a's and b's and will also be in S
  - $C(abaaba) = aabaaba$
  - $C(bbb) = abbb$

Function Machines

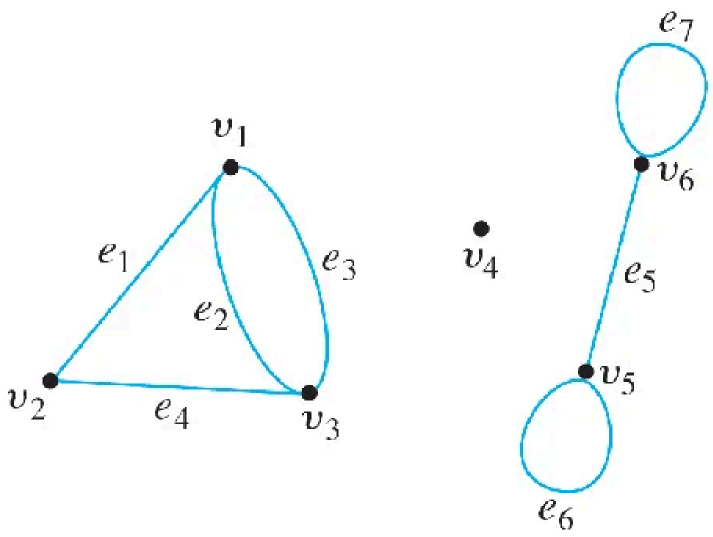
- Functions can be thought of as a machine that accepts some input x and responds with some output y

▼ [1.4] Language of Graphs

- **Graphs** are comprised of 2 finite sets: a nonempty set of **vertices** and a nonempty set of **edges** where each edge is associated with a set of one or two vertices known as its **endpoints**
- **Edge-endpoint functions** are the correspondence between edges and the sets containing their endpoints
- **Loops** are edges with one endpoint
- Two or more distinct edges with the same set of endpoints are **parallel**
- Since edges **connect** their endpoints:
  - Two vertices connected by the same edge are **adjacent**
    - A single vertex that is the endpoint of a loop is adjacent to itself
- Edges are **incident** on each of its endpoints
  - Two edges incident on the same endpoint are **adjacent**
  - A vertex where no edges are incident are **isolated**
- The **degree** of a vertex is the number of edges that are connected to it



Example



- Vertex set: $\{v_1,\ v_2,\ v_3,\ v_4,\ v_5,\ v_6\}$
- Edge set: $\{e_1,\ e_2,\ e_3,\ e_4,\ 3_5,\ e_6,\ e_7\}$
- Edge-endpoint function:

Edge	**Endpoints**
e_1	$\{v_1,\ v_2\}$
e_2	$\{v_1,\ v_3\}$
e_3	$\{v_1,\ v_3\}$
e_4	$\{v_2,\ v_3\}$
e_5	$\{v_5,\ v_6\}$
e_6	$\{v_5\}$
e_7	$\{v_6\}$

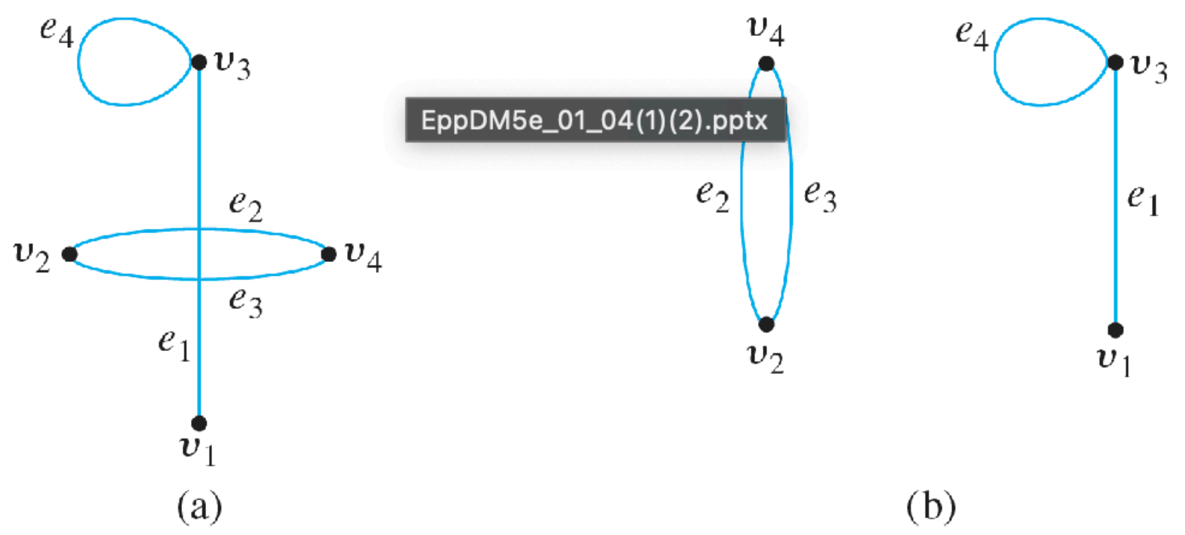
- The edges incident on v_1 are e_1, e_2, and e_3
- The vertices adjacent to v_1 are v_2 and v_3
- The edges adjacent to e_1 are e_2, e_3, and e_4
- The loops in the graph are e_6 and e_7
- The parallel edges in the graph are e_2 and e_3
- The vertices adjacent to themself are v_5 and v_6
- The only isolated vertex is v_4

Multiple Pictorial Representations

- Vertex set: $\{ v_1,\ v_2,\ v_3,\ v_4 \}$
- Edge set: $\{ e_1,\ e_2,\ e_3,\ e_4 \}$
- Edge-endpoint function:

Edge	Endpoints
e_1	$\{ v_1,\ v_3 \}$
e_2	$\{ v_2,\ v_4 \}$
e_3	$\{ v_2,\ v_4 \}$
e_4	$\{ v_3 \}$

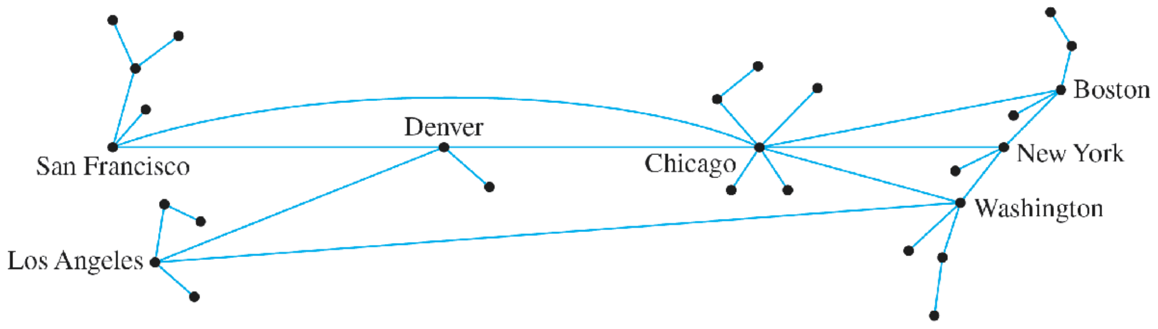
- Both figure (a) and figure (b) are pictorial representations of the aforementioned graph



- This can also be done in reverse by labelling two different drawings such that they represent the same graph

Graph Representations

- Many systems can be represented by graphs, with specific connecting edges being chosen to minimize costs, optimize services, etc
- A typical network, called a **hub-and-spoke model**, is show below:



- **Directed graphs/diagraphs** are graphs consisting of two finite sets including a nonempty set of vertices and a set of directed edges where each directed edge is associated with an ordered pair of vertices known as its endpoints
    - If edge e is associated with the pair (u, w) of vertices, then e is said to be the **(directed) edge** from u to w

[10] Theory of Graphs and Trees

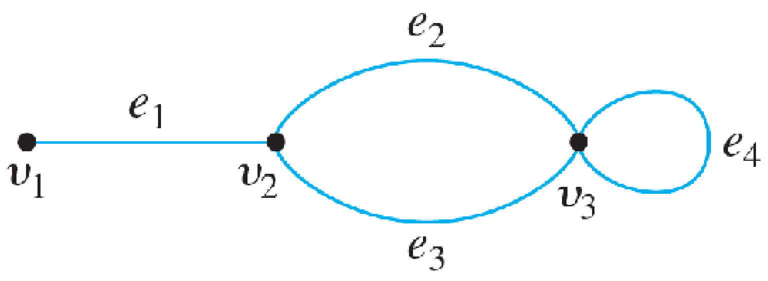
[10.1] Trails, Paths, and Circuits

- Graph theory was implemented by Euler to solve a problem concerning the possibility of someone walking across 7 bridges and ending in the exact same spot that they started in
- A **walk from v to w** is a finite alternating sequence of adjacent vertices and edges of a graph. Thus, a walk is represented by $v_0e_1v_1e_2\ldots v_{n-1}e_nv_n$
    - **Trails** are walks with no repeated edges
        - **Path** are trails with no repeated vertices
    - **Closed walks** are walks that start and end at the same vertex
        - **Circuit** are closed walks with at least one edge but no repeated edges
            - **Simple circuit** are circuits with no repeated vertices other than the start/end vertex
        - **Trivial walks** are walks consisting of just one vertex and no edges

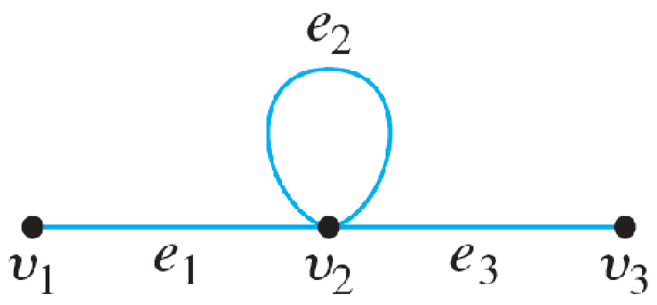
Notation for Walk

- The notation $e_1e_2e_4e_3$ can be used to describe a walk

- When accompanied by the graph below, it implicitly refers to the walk $v_1 e_1 v_2 e_2 v_3 e_4 v_3 e_3 v_2$
- Otherwise, using vertices can be important if you want to define a specific walk as there could be many ways those instructions with just edges could be interpreted

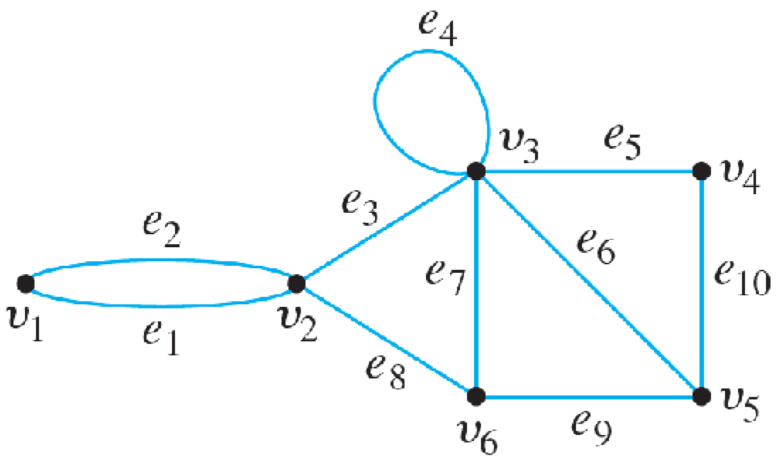


- The notation $v_2 v_3$ can also be used to describe a walk with the graph below, but without it the description is far more up to interpretation



Notation and Types of Walks

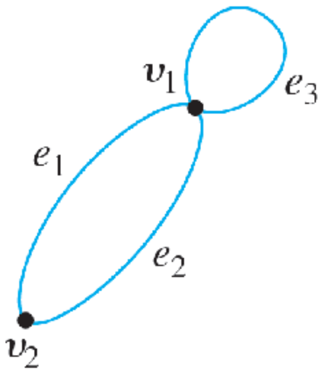
- Using the graph below and a few given notations, certain types of walks can be determined



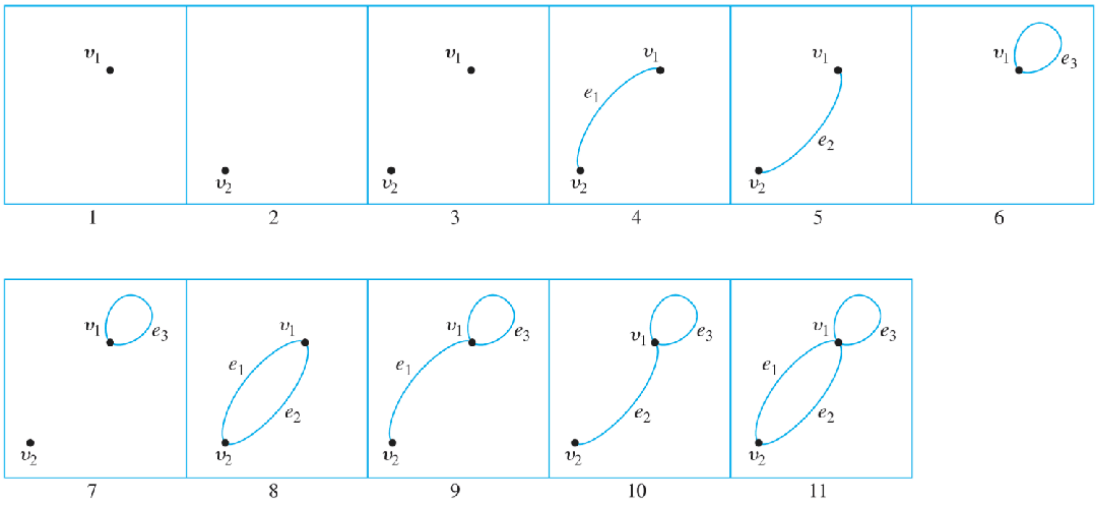
- $v_1e_1v_2e_3v_3e_4v_3e_5v_4$ is a trail from v_1 to v_4
- $e_1e_3e_5e_5e_6$ is a walk from v_1 to v_5
- $v_2v_3v_4v_5v_3v_6v_2$ is a circuit
- $v_2v_3v_4v_5v_6v_2$ is a simple circuit
- $v_1e_1v_2e_1v_1$ is a closed walk
- v_1 is a closed walk

Subgraph

- **Subgraph** are graphs whose vertices, edges, and endpoints are also within another graph
- For instance, given a graph G with the sets $\{v_1, v_2\}$ and $\{e_1, e_2, e_3\}$ with e_1 and e_2 being incident on v_1 and v_2 and e_3 looping at v_1, list all subgraphs

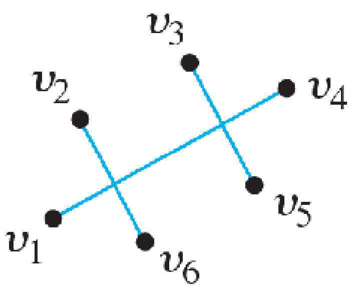


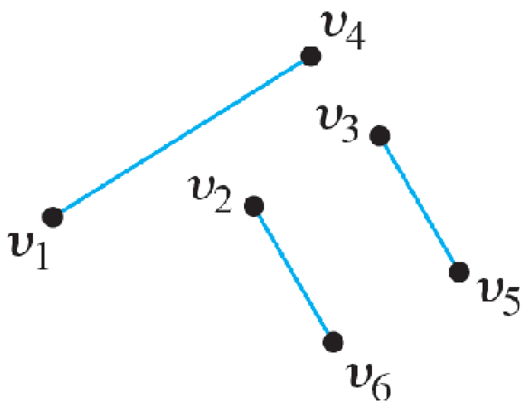
- There are 11 subgraphs of G, and they can be grouped by a lack of edges, having one edge, having two edges, and having three edges



Connectedness

- Graphs are connected if it is possible to travel from any vertex to any other adjacent vertex
  - Thus, a graph is not connected if there are at least two vertices not connected by any walk
- Let G be a graph
  - If G is connected, then any two distinct vertices of G can be connected by a path
  - If vertices v and w are part of a circuit in G and one edge is removed from the circuit, then there still exists a trail from v to w in G
  - If G is connected and G contains a circuit, then an edge of the circuit can be removed with disconnecting G
- For instance, the graph below on the left is not connected because there is no **direct path** between v_2 and v_1 and v_3 and v$_4$ which can be better illustrated by the graph on the right

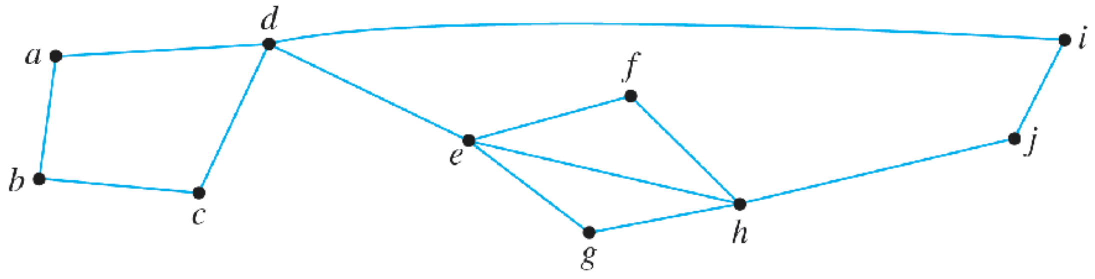




- **Connected component** → A connected subgraph of the largest possible size
- A graph H is a **connected component** of a graph G if, and only if
  - H is a subgraph of G
  - H is connected
  - No connected subgraph of G has H as a subgraph and contains vertices or edges not in H

Euler Trails and Circuits

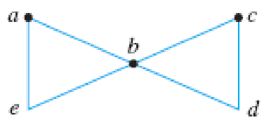
- **Euler trails** are sequences of adjacent edges and vertices that start at v, end at w, pass through every vertex at least once, and traverse every edge of the graph at least once
- **Euler circuit** are sequences of adjacent vertices and edges in a graph with at least one edge, start and end at the same vertex, use every vertex at least once, and use every edge exactly once
  - If a graph has a Euler circuit, then every vertex of the graph has positive even degree
  - If some vertex of a graph has an odd degree, then the graph does not have an Euler circuit
  - If all vertices of a connected graph are positive even integers, then the graph has a Euler circuit



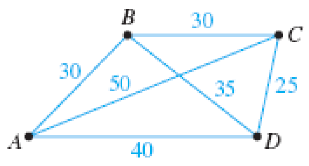
- Every vertex in the connected graph above has a degree of either 2 or 4, meaning that it has an Euler circuit

Hamiltonian Circuit

- **Hamiltonian circuits are** proposed circuits that can be taken by moving through the all of the vertices of a dodecahedron at least once while returning to the starting location
- If a graph G has a Hamiltonian circuit, then it has a subgraph H with these properties
  1. H contains every vertex of G
  2. H is connected
  3. H has the same number of edges as vertices
  4. Every vertex in H has a degree of 2
- In the following graph, this can be used to show that it cannot have a Hamiltonian circuit



- Notably, vertex b has a degree of 4, meaning that two edges incident on it must be removed. However, if that happens, there will be less edges than vertices and one of the corner vertices will have a degree of just 1
- The t**raveling salesman problem** is a famous problem based on finding the shortest possible Hamiltonian circuit needed for a salesman to travel to all cities at least once before returning home



- The best solution is to list out the possible Hamiltonian circuits with their corresponding total distances, noting that reverse routes have the same distance. In this case, $ABCDA$ is the best route
    - This problem is infamous for being factorial based, with medium-sized versions taking millions of years to finish the computation

▼ [10.2] Matrix Representations of Graphs

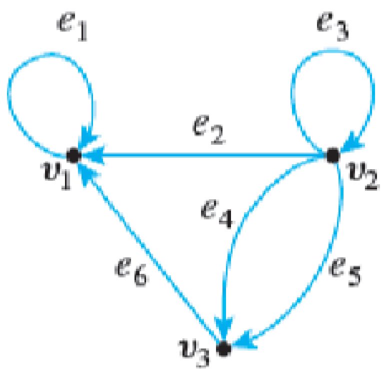
- Matrices are two dimensional analogues of sequences. They are also known as two dimensional arrays
- The following graph can be represented by the matrix $A = a_{ij}$ for which a_{ij}
    - ith row and jth column
- For instance, $m \times n$ matrix A over set S is an array with m rows and n columns

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix}$$

- If A and B are matrices, then $A = B$ if they are the same size and all of their corresponding entries are equal
    - $a_{ij} = b_{ij}$ for every $i = 1, 2, \dots, m$ and $j = 1, 2, \dots, n$
- **Square matrix** → A matrix square
    - **Main diagonal** → Represented by $a_{11}, a_{22}, \dots, a_{nn}$ for square matrix A of size $n \times n$

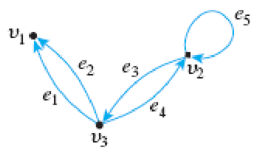
Matrices and Directed Graphs

- Directed graphs can be represented by matrices

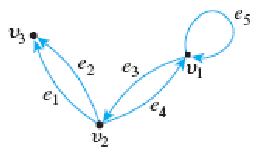


$$\mathbf{A} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 2 \\ 1 & 0 & 0 \end{bmatrix} \end{array}$$

- Each element a_{ij} in A represents the number of arrows from v_i to v_j
- **Adjacency matrices** are arrays representing a directed graph
- Nonzero entries on the main diagonal indicate loops
- Off-diagonal entries larger than 1 indicate parallel edges
- For example, these are two directed graphs that only differ by the order of vertices. Thus, their adjacency matrices will be different



(a)



(b)

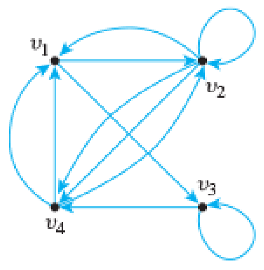
$$\mathbf{A}_a = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 2 & 0 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{A}_b = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

- Since there are 3 vertices in each graph, they are represented by 3×3 matrices
- Square matrices with nonnegative entries can also be used to describe directed graphs, but not unique ones as there are no information on edges

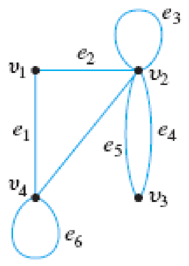
$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

- The matrix above produces the following directed graph



Matrices and Undirected Graphs

- Similar to the previous concept but each element a_{ij} in A represents the number of edges connecting to other ith and jth vertices of the graph
- Notably, for all values of i and j, element $a_{ij} = a_{ji}$, since adjacency matrices for undirected graphs are always **symmetric**
    - This is because there is no direction to the connections
- These graphs are similar, just lacking arrows for direction

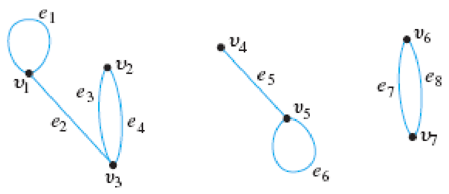


- This yields the following adjacency matrix

$$\mathbf{A} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 2 & 1 \\ 0 & 2 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

Matrices and Connected Components

- Graphs with several connected components have much different adjacency matrices



$$\mathbf{A} = \left[\begin{array}{ccc:cc:cc} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \end{array}\right]$$

- Notably, the adjacency matrix of the graph above is made up of multiple square matrices (of varying sizes) along its main diagonal
- Conversely, the other squares consist of 0s, as the connected components are all separated from each other and thus cannot be connected to each other's vertices
    - Sometimes, the other matrices are shown to not have any numbers at all and are just considered to be 0 since the vertices in those rows and columns have no connection
- For any graph G with connecting components G_i where i represents the number of n_i vertices in each component, the following is G's adjacency matrix

$$\begin{bmatrix} A_1 & O & O & \dots & O & O \\ O & A_2 & O & \dots & O & O \\ O & O & A_3 & \dots & O & O \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ O & O & O & \dots & O & A_k \end{bmatrix}$$

Matrix Multiplication

- Compared to the multiplication of real numbers, the matrices being multiplied together must be compatible (the number of rows in the first matrix must be equal to the number of columns in the second matrix and vice versa
- Additionally, matrix multiplication is not commutative, meaning that the order of the matrices in the operation actually matters
- The process below is how rows and columns are multiplied together

$$\begin{bmatrix} a_{i1} & a_{i2} & \dots & a_{in} \end{bmatrix} \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{bmatrix} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$$

- The visual below is a full demonstration of finding the product of two matrices after finding the dot product between each corresponding row and column
  - The resulting product of an $i \times j$ matrix and a $m \times n$ matrix is a $i \times n$ matrix

$$\mathbf{AB} = \begin{bmatrix} 2 & 0 & 3 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 4 & 3 \\ 2 & 2 \\ -2 & -1 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

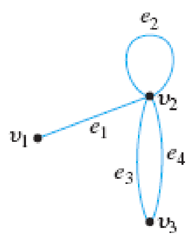
$$\begin{aligned} c_{11} &= 2 \cdot 4 + 0 \cdot 2 + 3 \cdot -2 = 8 + 0 - 6 = \mathbf{2} \\ c_{12} &= 2 \cdot 3 + 0 \cdot 2 + 3 \cdot -1 = 6 + 0 - 3 = \mathbf{3} \\ c_{21} &= -1 \cdot 4 + 1 \cdot 2 + 0 \cdot -2 = -4 + 2 + 0 = -\mathbf{2} \\ c_{22} &= -1 \cdot 3 + 1 \cdot 2 + 0 \cdot -1 = -3 + 2 + 0 = -\mathbf{1} \end{aligned}$$

$$\mathbf{AB} = \boxed{\begin{bmatrix} 2 & 3 \\ -2 & -1 \end{bmatrix}}$$

- **Identity matrix** → An $n \times n$ size matrix whose main diagonal is just 1s with all other numbers in the matrix being 0. It acts like multiplying by 1 with real numbers and is denoted by I
  - Any square matrix to the power of 0 gives identity matrix I

Counting Walks of Length N

- A walk in a graph consists of an alternating sequence of vertices and edges
  - If there are repeated edges being counted, then the number of edges in the sequence is the walk's **length**
- The walk $v_2e_3v_3e_4v_2e_2v_2e_3v_3$ has a length of 4
- The graph below can be used to define an example of how to find these walks



- If you want to find walks of length 2 from v_2 to v_2, you can use a second vertex as a starting point such as v_1 or v_3 or even v_2 itself by using the loop
    - In this case, there are 6 distinct walks
- Matrix multiplication can be used to determine walks of any length n based on matrix A^n
    - Squaring an adjacency matrix for instance can answer the number of walks of length 2
    - They can be used to multiply a particular column and row together (based on the desired connection) to answer the amount of walks of a particular length

$$\mathbf{A} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{A}^2 = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 6 & 2 \\ 2 & 2 & 4 \end{bmatrix}$$

- As seen above, the value in the second row and column is 6 of the adjacency matrix to the power of 2
    - This corresponds with 6 distinct walks from v_2 to v_2

$$\begin{bmatrix} \text{the first term} \\ \text{of this sum} \end{bmatrix} = \begin{bmatrix} \text{number of} \\ \text{edges from} \\ v_2 \text{ to } v_1 \end{bmatrix} \cdot \begin{bmatrix} \text{number of} \\ \text{edges from} \\ v_1 \text{ to } v_2 \end{bmatrix} = \begin{bmatrix} \text{number of pairs} \\ \text{of edges from} \\ v_2 \text{ to } v_1 \text{ and } v_1 \text{ to } v_2 \end{bmatrix}$$

▼ [10.3] Isomorphisms of Graphs

- Two graphs are **isomorphic** when they have the same connectivity but are graphically different

    - The two graphs would match up by having the edges between their corresponding vertices correspond to each other

- Properties can be **invariant** if they remain unchanged when a graph is relabelled
    - For an isomorphism, these are the invariant properties that is has where n, m, and k are nonnegative integers
        1. n vertices
        2. m edges
        3. Vertex with degree k
        4. m vertices of degree k
        5. Circuit of length k present
        6. Simple circuit of length k present
        7. m simple circuits of length k
        8. Connected
        9. Euler circuit present
        10. Hamiltonian circuit present

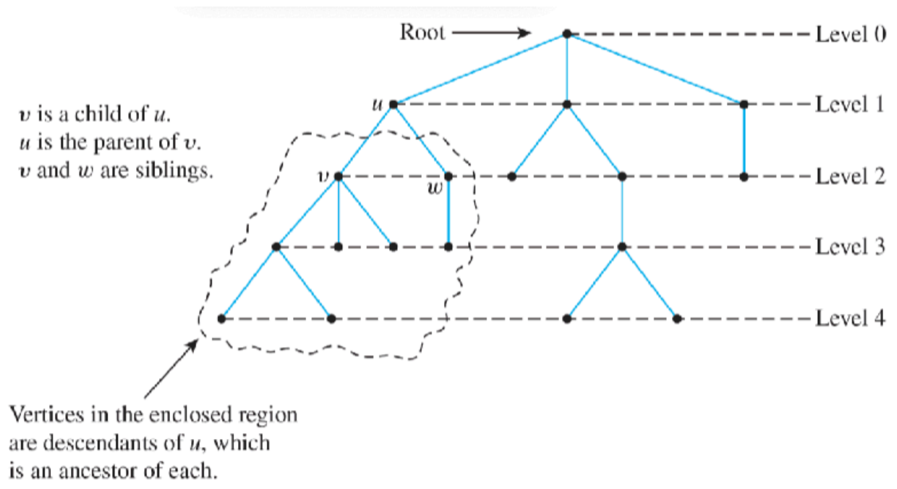
▼ [10.4] Trees: Examples and Basic Properties

- Graphs are **circuit-free** if it has no circuits meaning that they do not loop
- **Trees** are graphs that are both circuit-free and connected
- **Forest** are graphs that are circuit-free but not connected
- A common application is as a decision tree where decisions branch out
- A **parse tree** or **syntactic derivation tree** show the derivation of grammatically correct sentences from basic rules
    - The rules of grammar are called **productions**
    - The shorthand notation used for those rules is **Backus-Naur notation**
        - For instance, | represents or and <> encloses to be defined terms
    - **Semantics** are the meanings of words and their interrelations
    - **Syntax** is grammatical sentence structure

▼ [10.5] Rooted Trees

- **Rooted trees** are tree where one vertex is distinguished from the others, being known as the **root**

- Given any other vertex v in the tree, there is a unique path from, the root to v
- The number of edges in this path is called the **level** of v
  - The **height** is the length of the longest path
- The **root** is generally located at the top of the tree



A Rooted Tree

Binary Trees

- **Binary trees** are rooted trees where vertices have at most 2 children thus each child can be designated as the **unique** left or right child
- **Full binary trees** are binary trees where every parent has exactly two children
- They are used to represent algebraic expressions and the arbitrary nesting of parentheses
- If the number of internal vertices of a full binary tree is known, then the total number of vertices and leaves can be found
  - A full binary tree with k internal vertices has a total of $2k + 1$ vertices and $k + 1$ leaves
- Additionally, there is a maximum number of leaves for a given height h
  - A binary tree of height h has at most 2^h leaves and conversely a binary tree with t leaves has a height of at most $\log_2 t$

Binary Search Trees

- **Binary search trees** are a type of binary tree where data records can be stored, searched through, and processed efficiently

    - Records place into them must be in total order
    - The **key** is an element of a totally ordered set that can be added to each record and used to access each one
  - To build one, make the root and insert a key
  - To add a new node with a new key, compare it to the key at the root
    - If the new key is less than the key at the root, create a left child and insert a new key into it
    - If the new key is greater, create a right child and insert a new key into it
  - After a few keys have been added, you have to work down the tree from the root to find places to put new keys
  - For example, given keys 15, 10, 19, 25, 12, and 4 you would insert 15 as the root and then add left and right children based on the order of the keys

Additional Topics

- The **handshaking lemma** states that the sum of the degrees of all vertices in a finite graph is equal to twice the number of edges
- The **pigeonhole principle** states that if there are more pigeons that pigeonholes, then at least one pigeonhole will have more than one pigeon inside of it
  - This can be used to prove that all finite simple graphs have at least two vertices with the same degree if the pigeons are particular vertices and the pigeonholes are particular degrees

[2] The Logic of Compound Statements

▼ [2.1] Logical Form and Logical Equivalence

- An argument is a sequence of statements meant to demonstrate the truth of an assertion
- **Conclusions** are assertions at the end of a sequence
- **Premises** are statements preceding the conclusion
- Logic is a science of reasoning as it is more so meant to determine whether or not the conclusion and premises agree
- The common form of arguments is: “If p or q, then $\therefore$ if not r, the not p and not q”
  - If a bell rings or the flag drops, then the race is over. $\therefore$ If the race is not over, then the bell hasn’t run and the flag hasn’t dropped.

- P: (If logic is easy) or Q, then R. Q: (I will study hard). Therefore, R: (I will get an A) in this course.

Statements

- A **statement/proposition** is a sentence that is true or false but not both
  - $2+2=4$ is a true statement and $2+2=5$ is a false statement
  - $x^2+2=11$ depends on the value of x, so it is true for $x=3$ and $x=-3$ but false for other values of x

Compound Statements

- **Compound statements** are the combination of multiple simple statements using logical connectives, which are each representative of a particular logical phrase

$$\sim \text{ or } \neg \text{ is a unary operator representing "not"}$$

$$\wedge \text{ is a binary operator representing "and"}$$

$$\vee \text{ is a binary operator representing "or"}$$

- By letting simple statements be represented by statement variables, they can form boolean expressions with logical connectives connecting them
- For instance, let h be “it is hot” and s be “it is sunny”
  - “It is not hot but it is sunny.”
    - $\neg h \wedge s$
  - "It is neither hot nor sunny.”
    - $\neg h \wedge \neg s$
- This can also be done with inequalities as well. In this case, let p be $0<x$, q be $x<3$, and r be $x=3$
  - $x \leq 3$
    - $q \vee r$
  - $0<x<3$
    - $p \wedge q$
  - $0<x \leq 3$

        - $p \wedge (q \vee r)$

Truth Values

- The **negation** of p is "not p" or "it is not the case that p" and denoted as $\neg p$
- If p is true, $\neg p$ is false and vice versa
- **Conjunction** of p and q is "p and q" or $p \wedge q$
- **Disjunction** of p and q is "p or q" or $p \vee q$

Evaluating the Truth of More General Compound Statements

- **Statement form/propositional form** is an expression made from statement variables and logical connectives that become a statement when actual statements are substituted in
- **Truth tables** are tables showing truth values that correspond to combinations of truth values for component statement variables
- The table for $(p \vee q) \wedge \neg(p \wedge q)$is made up of numerous columns for each component

p	q	$p \vee q$	$p \wedge q$	$\neg(p \wedge q)$	$(p \vee q) \wedge \neg(p \wedge q)$
T	T	T	T	F	F
T	F	T	F	T	T
F	T	T	F	T	T
F	F	F	F	T	F

Logical Equivalence

- Statements are **logically equivalent** if, and only if, they have identical truth values for each possible substitution of statements for their statement variables. The logical equivalence of statement forms P and Q is denoted by writing $P \equiv Q$
- To test logical equivalence between P and Q
    1. Construct a truth table with a column for the truth values for each variable
    2. Check all combinations of truth values to see if the truth values of P and Q are the same
        a. If the truth value of P and Q is the same in each row, then they are logically equivalent
        b. Otherwise, they are not logically equivalent
- For instance, a table can be used to prove $\neg(p \wedge q) \equiv \neg p \wedge \neg q$

p	q	$\neg p$	$\neg q$	$p \wedge q$	$\neg(p \wedge q)$	$\neg p \wedge \neg q$
T	T	F	F	T	F	F
T	F	F	T	F	T	F
F	T	T	F	F	T	F
F	F	T	T	F	T	T

Different truth values, $\therefore$ not logically equivalent.

- Statements may also be used to prove $\neg(p \wedge q) \equiv \neg p \wedge \neg q$
  - Let p be the statement "$0 < 1$" and q be the statement "$1 < 0$"
  - $\neg(p \wedge q)$ is "It is not the case that both $0 < 1$ and $1 < 0$", this is true
  - On the other hand, $\neg p \wedge \neg q$ is "$0 \not< 1$ and $1 \not< 0$", this is false
- **De Morgan's Laws** → The negation of an "and" statement is equivalent to an "or" statement where each component is negated and vice versa

$$\neg(a \wedge b) \equiv \neg a \vee \neg b$$

$$\neg(a \vee b) \equiv \neg a \wedge \neg b$$

  - "John is 6 feet tell and he weighs at least 200 pounds" is logically equivalent to "John is not 6 feet tall or he weighs less than 200 pounds"
  - $-1 < x \leq 4$ is logically equivalent to $-1 \geq x$ or $x > 4$

Tautologies and Contradictions

- **Tautology** → A statement form that is always true regardless of the truth values of the individual statements substituted for its statement variables
  - **Tautological statement** → A statement whose form is tautology

$$\mathbf{t} \equiv p \vee \neg p$$

p	$\neg p$	$p \vee \neg p$
T	F	T
F	T	T

- **Contradiction** → A statement form that is always false regardless of the truth values of the individual statements substituted for its statement variables
  - **Contradictory statement** → A statement whose form is contradictory

$$\mathbf{c} \equiv p \wedge \neg p$$

p	$\neg p$	$p \wedge \neg p$
T	F	F
F	T	F

- Given $\mathbf{t}$ is a tautology and $\mathbf{c}$ is a contradiction, a truth table can be used to show $p \wedge \mathbf{t} \equiv \mathbf{t}$ and $p \wedge \mathbf{c} \equiv \mathbf{c}$

Negation Law

p	$\mathbf{t}$	$p \wedge \mathbf{t}$	p	$\mathbf{c}$	$p \wedge \mathbf{c}$
T	T	T	T	F	F
F	T	T	F	F	F

Logical Equivalence Summary

- Given any statement variables p, q, and r, and tautology $\mathbf{t}$ and contradiction $\mathbf{c}$, the following logical equivalences hold:

1. **Commutative laws:**

$$p \wedge q \equiv q \wedge p \quad \text{and} \quad p \vee q \equiv q \vee p$$

2. **Associative laws:**

$$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r) \quad \text{and} \quad (p \vee q) \vee r \equiv p \vee (q \vee r)$$

3. **Distributive laws:**

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r) \quad \text{and} \quad p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

4. **Identity laws:**

$$p \wedge \mathbf{t} \equiv p \quad \text{and} \quad p \vee \mathbf{c} \equiv p$$

5. **Negation laws:**

$$p \vee \neg p \equiv \mathbf{t} \quad \text{and} \quad p \wedge \neg p \equiv \mathbf{c}$$

6. **Double negative law:**

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

7. **Idempotent laws:**

$$p \wedge p \equiv p \quad \text{and} \quad p \vee p \equiv p$$

8. **Universal bound laws:**

$$p \vee \mathbf{t} \equiv \mathbf{t} \quad \text{and} \quad p \wedge \mathbf{c} \equiv \mathbf{c}$$

9. **De Morgan's laws:**

$$\neg(p \wedge q) \equiv \neg p \vee \neg q \quad \text{and} \quad \neg(p \vee q) \equiv \neg p \wedge \neg q$$

10. **Absorption laws:**

$$p \wedge (p \vee q) \equiv p \quad \text{and} \quad p \vee (p \wedge q) \equiv p$$

11. **Negations of $\mathbf{t}$ and $\mathbf{c}$:**

$$\neg\mathbf{t} \equiv \mathbf{c} \quad \text{and} \quad \neg\mathbf{c} \equiv \mathbf{t}$$

▼ [2.2] Conditional Statements

- **Conditional** ($\rightarrow$ or $\implies$) If p and q are statement variables, this represents "If p then q" or "implies q." It is true when p is true and implies q when it is true or whenever p is false
  - Denoted by:

$$p \implies q$$

  - p is the **hypothesis** or **antecedent**
  - q is the **conclusion** or **consequent**
- A conditional statement that is true by virtue of the fact that its hypothesis is false is often **vacuously true** or **true by default**
  - If $0 = 1$ then $1 = 2$ is a true statement because while the conclusion is not true, the hypothesis was already false
- The conditional operation holds the lowest priority

Given $p \vee \neg q \implies \neg p$

p	q	Conclusion $\neg p$	$\neg q$	Hypothesis $p \vee \neg q$	$p \vee \neg q \implies \neg p$
T	T	F	F	T	F
T	F	F	T	T	F
F	T	T	F	F	T
F	F	T	T	T	T

- When the hypothesis yields a false value, the value of the conclusion does not matter

Logical Equivalences Involving $\implies$

- Truth tables can be used to prove $p \vee q \implies r \equiv (p \implies r) \wedge (q \implies r)$
- The simplest way to do this is by filling in possible truth values for each side before comparing the results to see whether or not they are equivalent

p	q	r	$p \vee q$	$p \implies r$	$q \implies r$	$p \vee q \implies r$	$(p \implies r) \wedge (q \implies r)$
T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	F
T	F	T	T	T	T	T	T
T	F	F	T	F	T	F	F
F	T	T	T	T	T	T	T
F	T	F	T	T	F	F	F
F	F	T	F	T	T	T	T
F	F	F	F	T	T	T	T

- Since $p \vee q \rightarrow r$ and $(p \rightarrow q) \wedge (q \rightarrow r)$ have the same truth values, they are logically equivalent

Representation of If-Then as Or

$$p \implies q \equiv \neg p \vee q$$

- The following statement can be written in if-then form
  - "Either you get to work on time or you are fired."
    - This can be thought of as $p \implies q$ where p is "you get to work on time" and q is "you are fired"
  - "If you do not get to work on time then you are fired."

Negation of a Conditional Statement

- $p \implies q$ may only be false if the hypothesis p is true and the conclusion q is false
  - The negation of "if p then q" is logically equivalent to "p and not q"

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

- "If my car is in the repair shop, then I cannot get to class."
  - "My car is in the repair shop and I can get to class."
- "If Sara lives in Athens, then she lives in Greece."

    - "Sara lives in Athens and she does not live in Greece." (Sara might live in Athens, Georgia or Athens, Ohio)

- By applying DeMorgan's laws to the negation of the conditional statement, an "or" statement form of a conditional can be obtained

$$\neg\big(\neg(p \implies q)\big) \equiv \neg(p \wedge \neg q) \equiv \boxed{\neg p \vee q}$$

Contrapositive of a Conditional Statement

- The **contrapositive statement** of the conditional statement of the form "if p then q" is "if $\neg q$ then $\neg p$"
    - Symbolically:

$$\neg q \implies \neg p$$

    - Thus, a statement is logically equivalent to its contrapositive
- "If Howard can swim across the lake, then Howard can swim to the island."
    - "If Howard cannot swim to the island, then Howard cannot swim across the lake."
- "If today is Easter, then tomorrow is Monday."
    - "If today is not Monday, then today is not Easter."

The Converse and Inverse of a Conditional Statement

- Given the conditional statement "if p then q"
    - The **converse** is "If q then p"

$$q \implies p$$

    - The **inverse** is "If $\neg p$ then $\neg q$"

$$\neg p \implies \neg q$$

- "If Howard can swim across the lake, then Howard can swim to the island."
    - **Converse:** "If Howard can swim to the island, then Howard can swim across the lake."
    - **Inverse:** "If Howard cannot swim across the lake, then Howard cannot swim to the island."
- "If today is Easter, then tomorrow is Monday."
    - **Converse:** "If tomorrow is Monday, then today is Easter."

    - **Inverse:** “If today is not Easter, then tomorrow is not Monday.”
- A conditional statement is not logically equivalent with its converse and its inverse
- However, a conditional statement’s converse and inverse are logically equivalent to each other

Only If and Biconditional

- If p and q are statements, p **only if** q means “if not q then not p” or equivalently, “p then q”

$$\neg q \implies \neg p \equiv p \implies q$$

- “John will break the world’s record for the mile run only if he runs the mile in under four minutes”
    1. “If John does not run the mile in under four minutes, then he will not break the world’s record for the mile run.”
    2. “If John breaks the world’s record for the mile run, then he will have ran the mile in under four minutes.”
    - Statement 1 and 2 are contrapositives
- Given statement variables p and q, the **biconditional** of p and q is “p if, and only if, q”
    - This evaluates to true if p and q have the same truth values and false if they have different truth values
    - **iff** is shorthand for “if, and only if”

$$p \iff q$$

- Below is the truth table for a **biconditional**

p	q	$p \iff q$
T	T	T
T	F	F
F	T	F
F	F	T

- Order operations hierarchy
    1. $\neg$ Negations come first
    2. $\wedge, \vee$ Evaluate “and” and “or” next, parentheses for precedence
    3. $\implies$, $\iff$ Evaluate “conditionals” and “iff” next, parentheses for precedence

- Additionally, a biconditional is the equivalent of a conjunction of two converse conditionals (given the same two statement variables)
- "This computer program is correct if, and only if, it produces correct answers for all possible sets of input data"
    - "If this program is correct, then it produces the correct answers for all possible sets of input data; and f this program produces the correct answers for all possible sets of input data, then it is correct"
    - $p \implies q \land q \implies p$
- This shows $p \iff q \equiv (p \implies q) \land (q \implies p)$
- The following is the truth table for this logical equivalence

p	q	$p \implies q$	$q \implies p$	$(p \implies q) \land (q \implies p)$	$p \iff q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

Necessary and Sufficient Conditions

- If r and s are statements
    - r is a **sufficient condition** for s means that "if r then s"

$$r \implies s$$

    - r is a **necessary condition** for s means that "if not r, then not s"

$$\neg r \implies \neg s$$

- "Pia's birth on U.S. soil is a sufficient condition for her to be a U.S citizen."
    - "If Pia was born on U.S. soil, then she is a U.S. citizen."
- "George's attaining age 35 is a necessary condition for his being president of the United States."
    - "If George has not attained the age of 35, then he cannot be the president of the United States."
    - "If George can be the president of the United States, then he has attained the age of 35."
    - Contrapositive is relevant here as there are two ways to write the necessary condition

▼ [2.3] Valid and Invalid Arguments

- Mathematically, **arguments** are a sequence of statements ending with a conclusion
- In **argument form**, everything before the **conclusion** are the **premises/assumptions/hypotheses**
- The validity of an argument is based off of whether or not a conclusion follows necessarily from the preceding statements
- An argument form is **valid** when the premises are true, meaning that the conclusion is also true
  - An **argument** being valid means that its form is valid
- To test validity:
  1. Identify the premises and conclusion of the argument form
  2. Make a truth table of the truth values of the premises and conclusion
  3. **Critical rows** are rows where all premises are true
     a. If there are critical rows with a false conclusion value, then the arguments of the form may have true premises and a false conclusion, so the form is invalid
     b. If all critical rows have a true conclusion value, then the argument form is valid
- Below is an argument form represented abstractly

$$\begin{aligned} p &\implies q \vee \neg r && \text{(Premise)} \\ q &\implies p \wedge r && \text{(Premise)} \\ \therefore p &\implies r && \text{(Conclusion)} \end{aligned}$$

  - Constructing a truth table provides a clear way to view an argument's validity
  - Notice how the values of the conclusion do not matter in rows with at least one false premise
  - This argument form is invalid as seen below

						Premises		Conclusion
p	q	r	$\neg r$	$q \vee \neg r$	$p \wedge r$	$p \implies q \vee \neg r$	$q \implies p \wedge r$	$p \implies r$
T	T	T	F	T	T	T	T	T
T	T	F	T	T	F	T	F	
T	F	T	F	F	T	F	T	
T	F	F	T	T	F	T	T	F
F	T	T	F	T	F	T	F	
F	T	F	T	T	F	T	F	
F	F	T	F	F	F	T	T	T
F	F	F	T	T	F	T	T	T

Modus Ponens and Modus Tollens

- A **syllogism** is an argument form with two premises and a conclusion
  - The first and second premises are a **major premise** and **minor premise** respectively
- **Modus ponens** are a form of syllogisms in logic given in the following form

$$\begin{gathered}\text{If } p \text{ then q.}\\ p\\ \therefore q\end{gathered}$$

- "If the sum of the digits of 371,487 is divisible by 3, then 371,487 is divisible by 3"
  - "The sum of digits 371,487 is divisible by 3. ∴ 371,487 is divisible by 3"
- **Modus tollens** are another valid argument form, being the contrapositive of modus ponens

$$\begin{gathered}\text{If } p \text{ then q.}\\ \neg q\\ \therefore \neg p\end{gathered}$$

- "If 870,232 is divisible by 6, then it is divisible by 3. 870,232 is not divisible by 3. ∴ 870,232 is not divisible by 6"

Summary of Rules of Inference

- **Generalization**
- **Specialization**
- **Conjunction**
- **Elimination**

- **Transitivity**
- **Proof by Division into Cases**
- **Contradiction Rule**

Additional Valid Arguments Forms: Rules of Inference

- A **rule of inference** is a form of argument that is valid
  - For example, modus ponens and modus tollens are both rules of inference
- **Generalizations** are used to make generalizations, as if p is true, then generally p or some other statement q may be true
  - "Anton is a junior. $\therefore$ (more generally) Anton is a junior or Anton is a senior"

$$\begin{array}{cc} p & q \\ \therefore p \lor q & \therefore p \lor q \end{array}$$

- **Specialization** is useful for specifically describing objects by discarding irrelevant information
  - If you just want someone who knows graph algorithms:
  - "Ana knows numerical analysis and Ana knows graph algorithms. $\therefore$ (in particular) Ana knows graph algorithms"

$$\begin{array}{cc} p \land q & p \land q \\ \therefore p & \therefore q \end{array}$$

- **Elimination** is when one out of two possibilities can be ruled out, thus the other must be the case

$$\begin{array}{cc} p \lor q & p \lor q \\ \neg q & \neg p \\ \therefore p & \therefore q \end{array}$$

- **Transitivity** if one statement implies a second statement and the second implies a third, the first implies the third
  - For instance, if n is some integer
  - If n is divisible by 18, then n is divisible by 9

$$\begin{aligned} p &\implies q \\ q &\implies r \\ \therefore p &\implies r \end{aligned}$$

- **Proof by Division into Cases** is generally when the same conclusion is reached between numerous possibilities, meaning that the conclusion must be true
    - If x is positive or x is negative
    - If x is positive, then $x^2 > 0$
    - If x is negative, then $x^2 > 0$
    - $\therefore x^2 > 0$
    - In this case, x is known to be a nonzero real number, thus the sole conclusion is $x^2 > 0$

$$\begin{aligned} &p \lor q \\ &p \implies r \\ &q \implies r \\ \therefore\ &r \end{aligned}$$

- You are about to leave for class in the morning and discover that you don't have your glasses. Thus, you need to find where they are. You know the following statements are true:
    1. If I was reading my class notes in the kitchen, then my glasses are on the kitchen table.
    2. If my glasses are on the kitchen table, then I saw them at breakfast.
    3. I did not see my glasses at breakfast.
    4. I was reading my class notes in the living room or I was reading my class notes in the kitchen.
    5. If I was reading my class notes in the living room, then my glasses are on the coffee table.
    - Given statement 1 and statement 2, you can say "If I was reading my class notes in the kitchen, then I saw them at breakfast" by transitivity
    - Given statement 3, you can say "I did not see my glasses at breakfast. Therefore, I did not read my class notes in the kitchen" by modus tollens
    - Thus, in statement 4 you can say "I was reading my class notes in the living room" by elimination
    - As a result, in statement 5 you can say "I read my class notes in the living room. Therefore, they are on the coffee table" by modus ponens

Fallacies

- **Fallacies** are errors in reasoning that cause an argument to be invalid
- Common fallacies include:
  - **Ambiguous premises** that are treated as if they are unambiguous
  - **Circular reasoning** (assuming something is proved without proper derivation from premises)
  - **Jumping to conclusion** without adequate grounds
- **Converse error** is another fallacy, and stems from believing that something implied by another thing can be used to conclude that the other thing is true
  - "If Zeke is a cheater, then Zeke sits in the back row"
  - "Zeke sits in the back row"
  - "$\therefore$ Zeke is a cheater"
  - This line of reasoning clearly doesn't make sense
  - "If New York is a big city, then New York has tall buildings"
  - "New York has tall buildings"
  - "$\therefore$ New York is a big city"
  - In this example, the conclusion is true, but the line of reasoning still does not make sense

$$\begin{gathered} p \implies q \\ q \\ \therefore p \end{gathered}$$

p	q	$p \implies q$	q	p
T	T	T	T	T
T	F	F	F	
F	T	T	T	F
F	F	T	F	

- **Inverse error** is another fallacy, and stems from believing that the inverse of an implication would also be valid
  - "If there two vertices are adjacent, then they do not have the same color"
  - "These two vertices are not adjacent"
  - "$\therefore$ they have the same color"
  - As with the previous error, this doesn't make sense at all

$$\begin{array}{c} p \implies q \\ \neg p \\ \therefore \neg q \end{array}$$

p	q	$p \implies q$	$\neg p$	$\neg q$
T	T	T	T	F
T	F	F	T	
F	T	T	F	
F	F	T	F	

- Arguments are only **sound** if, and only if, it is valid and its premises are true
  - Otherwise, they are **unsound**

Contradictions and Valid Arguments

- **Contradiction rule** states that if an assumption that statement p is false leads to a contradiction, then p must be true
  - Thus, if an assumption leads to a contradiction, the assumption is false
- Given that **c** is a contradiction, prove the following argument form

$$\begin{array}{c} \neg p \implies \mathbf{c} \\ \therefore p \end{array}$$

			Premise	Concl.
p	$\neg p$	$\mathbf{c}$	$\neg p \implies \mathbf{c}$	p
T	F	F	T	T
F	T	F	F	

- Raymond Smullyan's example of an island containing either truth-telling knights or lying knaves can be used to show how the contradiction works
- A says: B is a knight
- B says: A and I are of an opposite type
  1. Suppose A is a knight
  2. $\therefore$ what A says is true
  3. $\therefore$ B is also a knight
  4. $\therefore$ what B says is true
  5. $\therefore$ A and B are of opposite types

6. ∴ there is a contradiction: A and B are both knights and A and B are of an opposite type
7. ∴ the supposition is false
8. ∴ A is not a knight
9. ∴ A is a knave
10. ∴ what A says is false
11. ∴ B is not a knight
12. ∴ B is also a knave
13. ∴ both A and B are knaves

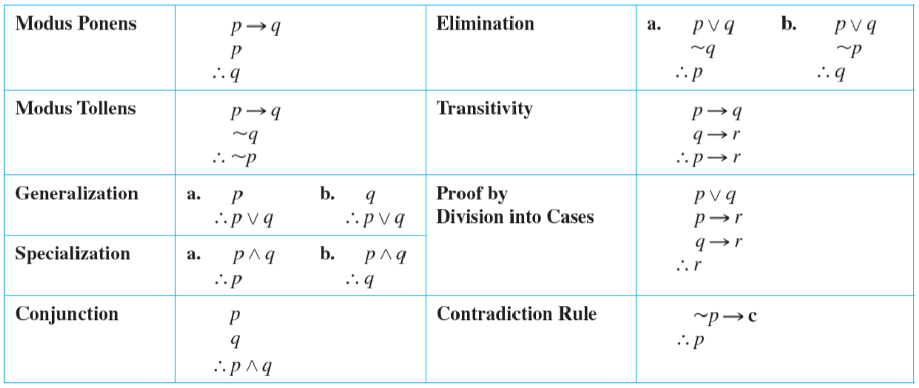
- Here is another example with more variables
    - U says: None of us is a knight
    - V says: At least three of us are knights
    - W says: At most three of us are knights
    - X says: Exactly five of us are knights
    - Y says: Exactly two of us are knights
    - Z says: Exactly one of us is a knight
        1. Suppose U is a knight (1)
        2. ∴ what U says is true
        3. ∴ no one is a knight
        4. ∴ everyone is a knave
        5. ∴ there is a contradiction: U is a knight and everyone is a knave
        6. ∴ the supposition is false
        7. ∴ U is a not a knight
        8. ∴ U is a knave
        9. ∴ what U says is false
        10. ∴ at least 1 person is a knight
        11. Suppose V is a knight (2)
        12. ∴ what V says is true
        13. ∴ at least 3 people are knights

14. Suppose W is a knight (3)
15. ∴ what W says is true
16. ∴ at most 3 people are knights
17. Suppose X is a knight (4)
18. ∴ what X says is true
19. ∴ exactly 5 people are knights
20. ∴ there is a contradiction: exactly 5 people are knights and there is at most 3 knights and there is at least 3 knights and at least 1 person is a knight
21. ∴ the supposition is false
22. ∴ X is a knave
23. ∴ what X says is false
24. Suppose Y is a knight (5)
25. ∴ what Y says is true
26. ∴ Exactly 2 people are knights
27. ∴ there is a contradiction: exactly 2 people are knights and there is at least 1 knight and there is at least 3 knights and there is at most 3 knights
28. ∴ supposition 2 and 3 are false
29. ∴ V is a knave
30. ∴ what V says is false
31. Suppose Z is a knight
32. ∴ what Z says is true
33. ∴ there is exactly one knight
34. ∴ there is a contradiction: exactly 1 person is a knight and exactly 2 people are knights are there is at most 3 knights
35. ∴ the supposition is false
36. ∴ Z is a knave
37. ∴ what Z says is false
38. ∴ Y and W are knights and everyone else is a knave

- These previous examples are **conditional proofs**, meaning that an assumption is made that the antecedent is true before logically deriving the conclusion

- The assumption is discarded by the end as a true conclusion was reached when all the premises were true

Rules of Inference Summary



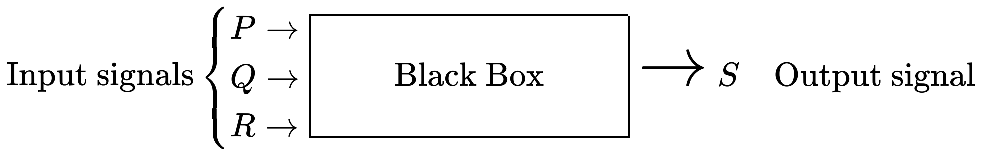
Modus Ponens	$p \to q$ p $\therefore q$	**Elimination**	**a.** $p \vee q$ $\sim q$ $\therefore p$ **b.** $p \vee q$ $\sim p$ $\therefore q$
Modus Tollens	$p \to q$ $\sim q$ $\therefore \sim p$	**Transitivity**	$p \to q$ $q \to r$ $\therefore p \to r$
Generalization	**a.** p $\therefore p \vee q$ **b.** q $\therefore p \vee q$	**Proof by Division into Cases**	$p \vee q$ $p \to r$ $q \to r$ $\therefore r$
Specialization	**a.** $p \wedge q$ $\therefore p$ **b.** $p \wedge q$ $\therefore q$		
Conjunction	p q $\therefore p \wedge q$	**Contradiction Rule**	$\sim p \to \mathbf{c}$ $\therefore p$

[2.4] Application: Digital Logic Circuits

- Input signals are either 0 or 1
  - They correspond to *false* or *true* respectively

Black Boxes and Gates

- Some engineers and digital system designers think of basic circuits as **black boxes**
- Black boxes are abstractions that don't show the implementation, but do show the inputs and outputs
  - Their operation is specified by the construction of an input/output table



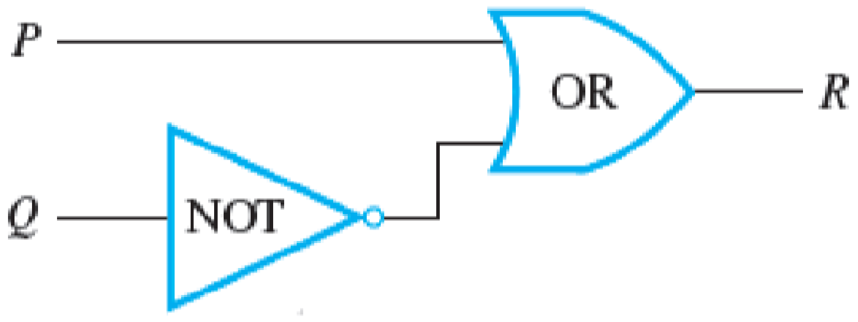
- In this case, there are a total of 2^3 or 8 combinations

P	Q	R
1	1	1
1	1	0
1	0	1
1	0	0
0	1	1
0	1	0
0	0	1
0	0	0

- A **NOT-gate** (or **inverter**) is a circuit with one input signal and one output signal that inverts an input of 0 to 1 and an input of 1 to 0
    - An **AND-gate** is a circuit with two input signals and one output signal that only outputs a signal of 1 if both inputs are 1
- An **OR-gate** is a circuit with two input signals and one output signal that outputs a signal of 1 as long as at least 1 input has a signal of 1
- There is a set of rules for a combinational circuit
    1. Never combine two input wires
    2. A single input wire can be split partway and used as input for two separate gates
    3. An output wire can be used as an input
    4. No output of a gate can eventually feed back into that gate
    - The last rule can be broken in complex circuits known as **sequential circuits** whose output at any time depends on both current and previous inputs

The Input/Output Table for a Circuit

- If you are given a set of input signals for a circuit, you can find its output by tracing through the circuit gate by gate

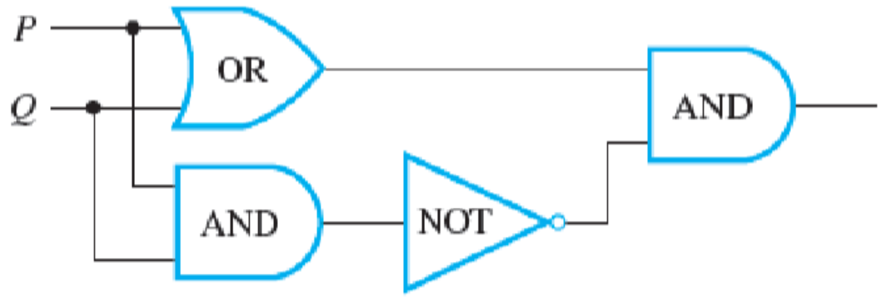


P	Q	R
1	1	1
1	0	1
0	1	0
0	0	1

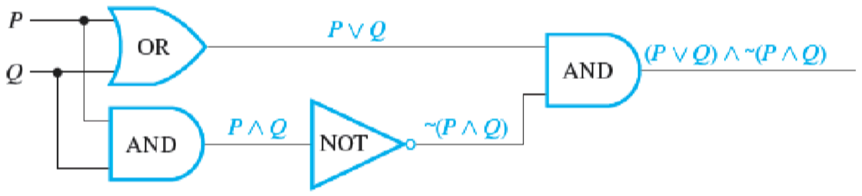
The Boolean Expression Corresponding to a Circuit

- Any variable (including statement variables or input signals) that can take one of only two values is a **Boolean variable**

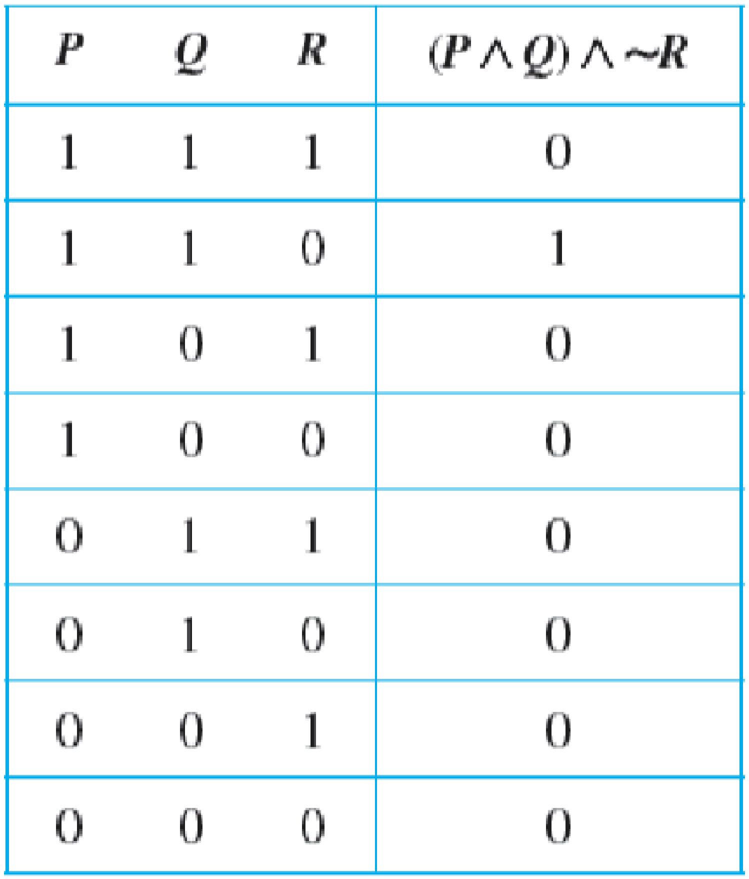
- A **Boolean expression** is an expression composed of Boolean variables and the connectives, $\neg$, $\wedge$, $\vee$
- Black dots refer to the soldering of two wires, thus overlaps without black dots do not count as connections
    - This corresponds to certain inputs being involved in numerous gates, as represented



- As a result, compound statements can be used to represent them



- This can be represented by: $(P \vee Q) \wedge \neg(P \wedge Q)$
- A **recognizer** is a circuit that outputs 1 for one specific combination of input signals and 0 for anything else



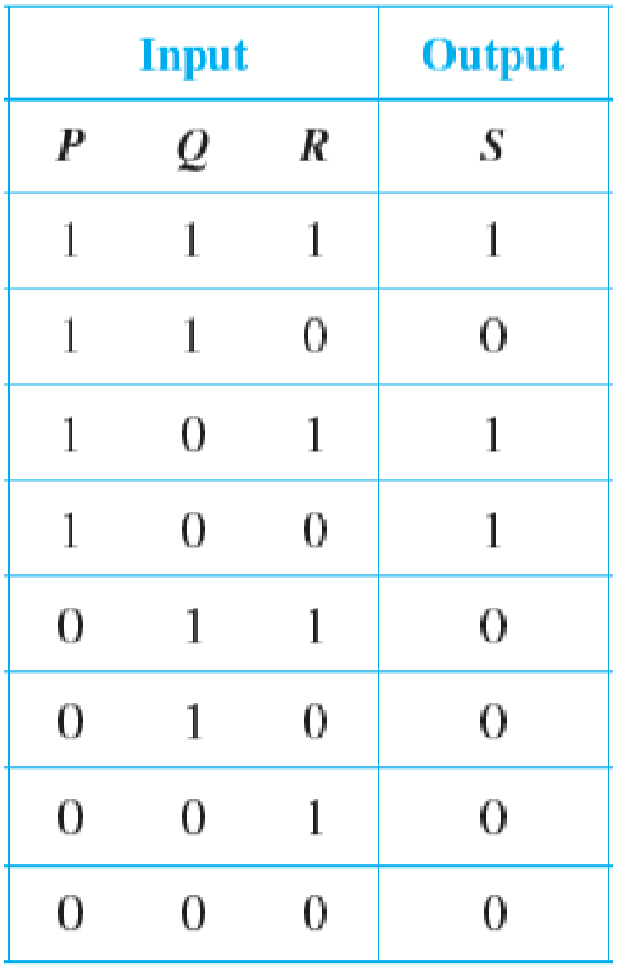
P	Q	R	$(P \wedge Q) \wedge \sim R$
1	1	1	0
1	1	0	1
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

The Circuit Corresponding to a Boolean Expression

- The best strategy is to work from the outermost part of the Boolean expression toward the innermost part
    - This allows logic gates to be added for each corresponding operation
- Operations with parentheses can be thought of as the result of one logic gate, thus they should be evaluated earlier in the circuit

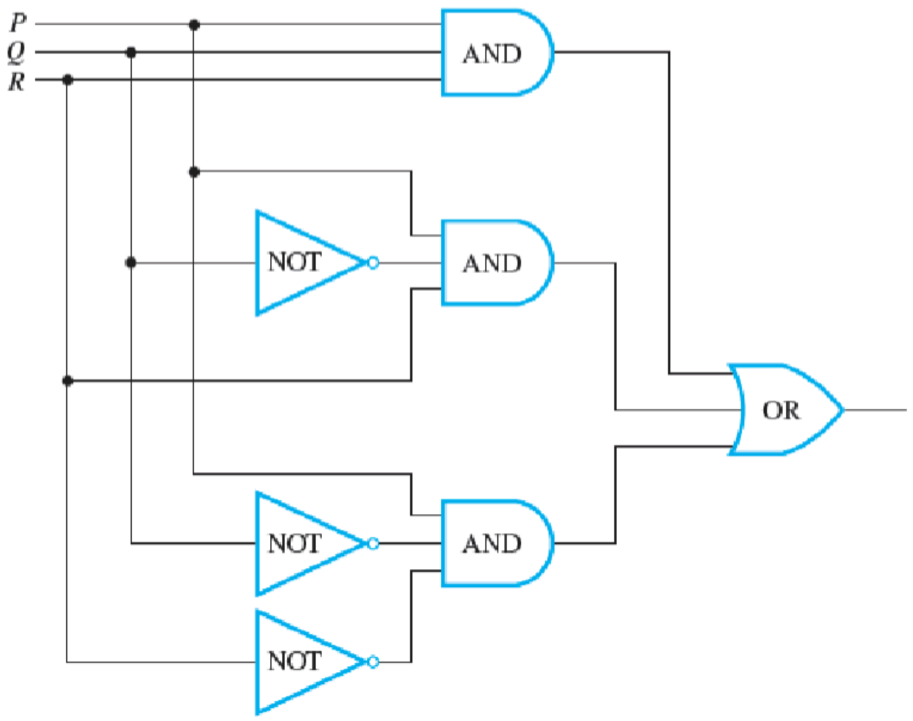
Finding a Circuit That Corresponds to a Given Input/Output Table

- The best strategy here is to construct a Boolean expression using the table before using the previous strategy to convert it into a circuit



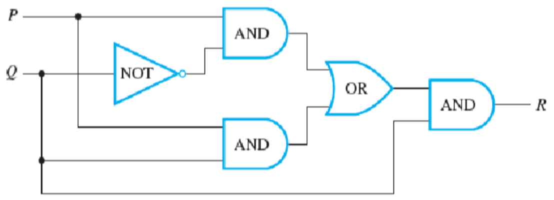
Input			Output
P	Q	R	S
1	1	1	1
1	1	0	0
1	0	1	1
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

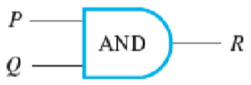
- For instance, the best place to start here is to look at the rows with an output signal of 1
    - In this case, row 1 can be associated with $P \wedge Q \wedge R = 1$
    - Row 3 could be associated with $P \wedge \neg Q \wedge R = 1$
    - Row 4 could be associated with $P \wedge \neg Q \wedge \neg R = 1$
    - Combining them, the Boolean expression for the truth table is $(P \wedge Q \wedge R) \vee (P \wedge \neg Q \wedge R) \vee (P \wedge \neg Q \wedge \neg R)$
        - Notably, the expression is a disjunction of terms that are themselves conjunctions, known as a **disjunctive normal form** or **sum-of-products form**



Simplifying Combinational Circuits

- The following circuits below differ greatly in complexity, but actually lead to the same truth table
- Thus, it makes sense to try to simplify circuits to use





- To simplify the first circuit to the second circuit, it can be represented a Boolean expression
    1. $\big((P \wedge \neg Q) \vee (P \wedge Q)\big) \wedge Q$
    2. $\equiv \big(P \wedge (Q \vee \neg Q)\big) \wedge Q$ Distributive law
    3. $\equiv (P \wedge \mathbf{t}) \wedge Q$ Negation law
    4. $\equiv P \wedge Q$ Identity law

- Two digital logic circuits are **equivalent** if, and only if, their input/output tables are identical

NAND and NOR Gates

- A **NAND-gate** is a single gate that acts like an AND-gate followed by a NOT-gate
    - The output signal is 1 only when at least one input signal is 0
    - They are denoted by the **Sheffer stroke** $|$, although it may also be an **upward pierce arrow** $\uparrow$

Input		Output
P	Q	$R = P \mid Q$
1	1	0
1	0	1
0	1	1
0	0	1

- Related idempotent proof

$$\neg P \equiv P \mid P?$$

$$\begin{aligned} \neg P &\equiv \neg(P \wedge P) && \text{(Idempotent Laws)} \\ &\equiv P \mid P && \text{(Def of Sheffer stroke)} \end{aligned}$$

- A **NOR-gate** acts like an OR-gate followed by a NOT-gate
    - The output signal is 1 only when both input signals are 0
    - They are denoted by the **downward pierce arrow** $\downarrow$

Input		Output
P	Q	$R = P \downarrow Q$
1	1	0
1	0	0
0	1	0
0	0	1

Extra information: XOR Gates

- **XOR** gates represent the **exclusive or** in logic (not covered in this course)
    - The output signal is 1 only if both inputs have different values
    - They are denoted by a circle around a plus sign $\oplus$

Input		Output
P	Q	$R = P \oplus Q$
1	1	0
1	0	1
0	1	1
0	0	0

▼ [2.5] Application: Number Systems and Circuits for Addition

Binary Representation of Numbers

- **Binary notation** or **base 2 notation** is important for modern electronics because 0 or 1 can be used to describe the two main states they may be in

$$\begin{aligned}
1_{10} &= 1 \cdot 2^0 && = 0001_2 \\
2_{10} &= 1 \cdot 2^1 + 0 \cdot 2^0 && = 0010_2 \\
3_{10} &= 1 \cdot 2^1 + 1 \cdot 2^0 && = 0011_2 \\
4_{10} &= 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 && = 0100_2 \\
5_{10} &= 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 && = 0101_2 \\
6_{10} &= 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 && = 0110_2 \\
7_{10} &= 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 && = 0111_2 \\
8_{10} &= 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 && = 1000_2 \\
9_{10} &= 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^1 && = 1001_2
\end{aligned}$$

- When converting from binary to decimal, convert each digit to decimal form and add them together
- When converting from decimal to binary, find the largest power of two, subtract it from the number, and add it. Keep doing this until you subtract it until 0

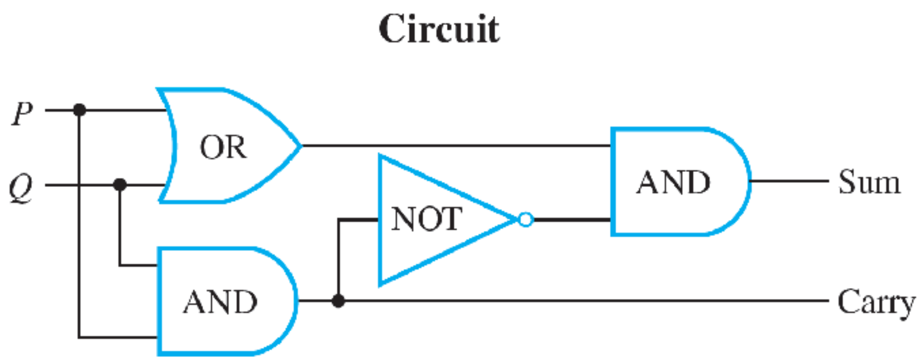
Binary Addition and Subtraction

- When adding or subtracting binary numbers, the number 2 should be treated as a 10 is in decimal notation (carry 1 over to next digit)

Circuits for Computer Addition

- If a circuit is designed to produce the sum of two binary digits P and Q, P and Q can be represented by a 0 or a 1
    - In this case, the circuit must have two outputs, one for the left digit (known as the **carry**) and one for the right digit (known as the **sum**)

- Since the carry output is 1 if both P and Q are 1 and 0 otherwise, it can be represented by an AND-gate or $P \wedge Q$
- Also, the sum output is 1 if only P or only Q is 1, which can be represented by an **XOR-gate** or $(P \vee Q) \wedge \neg(P \wedge Q)$
- This kind of circuit is known as a **half-adder**



- To construct a circuit that can add multi-digit binary numbers, it needs a circuit to compute the sum of three binary digits
  - This kind of circuit is known as a **full-adder**

$$\begin{array}{rccc} & & 1 & & \leftarrow \text{carry row} \\ & & 1 & 1_2 & \\ + & & 1 & 1_2 & \\ \hline & 1 & 1 & 0_2 & \end{array}$$

- Two full-adders and one half-adder can add two three-digit binary number

Two’s Complements and the Computer Representation of Signed Integers

- Binary addition can be complicated and 0s are not a unique representation, thus two’s complements are used due to having a unique 0 representation and are very easy to add
- Bit lengths of 64 are 32 are the most common, although this course will generally cover bit lengths of 8
- The 8-bit two’s complement for an integer a between -128 and 127 is the 8-bit binary representation for

$$\begin{cases} a & \text{if } a \geq 0 \\ 2^8 - |a| & \text{if } a < 0 \end{cases}$$

- Using the previous formula, -46 can be calculated as:

$$\begin{aligned}(2^8 - |-46|)_{10} &= (256 - 46)_{10} \\ &= 210_{10} \\ &= (128 + 64 + 16 + 2)_{10} \\ &= 11010010_2\end{aligned}$$

- The simplest method for determining the 8-bit two's complement for a negative integer is to write the 8-bit binary representation for $|a|$ before inverting all digits and adding 1 in binary
    - The inversion part can also be done by inverting everything to the left of the first 1

$$\begin{aligned}|-46|_{10} &= 46_{10} \\ &= (32 + 8 + 4 + 2)_{10} \\ &= 011110_2 \\ &= 100010_2\end{aligned}$$

- An 8-bit two's complement's range is limited due to the leftmost sign being used to signify sign (0 for positive, 1 for negative)
    - When operations are applied, always drop bits after the eighth one
- When finding the decimal representation of two's complement, determine the sign first using the leftmost digit before applying the two's complement and adding back the sign determined earlier

Addition and Subtraction with Integers in Two's Complement Form

- Using the two's complement form, subtraction can be simplified to adding a negative number in two's complement

$$a - b = a + (-b)$$

- After this, the resulting integers can be added together using ordinary binary addition while discarding carry bits of 1 in the 2^8th position and converting back to decimal form

Hexadecimal Notation

- **Hexadecimal notation** or **base 16 notation** is a compact notation that can be easily converted to binary when needed
- Compared to decimal there are 6 additional digits

$$
\begin{aligned}
10_{10} &= A_{16} \\
11_{10} &= B_{16} \\
12_{10} &= C_{16} \\
13_{10} &= D_{16} \\
14_{10} &= E_{16} \\
15_{10} &= F_{16}
\end{aligned}
$$

- Since there are 2^4 digits in hexadecimal, they can be represented by 4-bit binary clusters each, making them very easy to convert

$$
\begin{aligned}
0_{16} &= 0000 \\
1_{16} &= 0001 \\
2_{16} &= 0010 \\
3_{16} &= 0011 \\
4_{16} &= 0100 \\
5_{16} &= 0101 \\
6_{16} &= 0110 \\
7_{16} &= 0111 \\
8_{16} &= 1000 \\
9_{16} &= 1001 \\
A_{16} &= 1010 \\
B_{16} &= 1011 \\
C_{16} &= 1100 \\
D_{16} &= 1101 \\
E_{16} &= 1110 \\
F_{16} &= 1111
\end{aligned}
$$

- When converting from hexadecimal to decimal, multiply each consecutive digit by 16^n where n represents the place

$$
\begin{aligned}
3CF_{16} &= (3 \cdot 16^2 + 12 \cdot 16^1 + 15 \cdot 16^0)_{10} \\
&= (3 \cdot 256 + 12 \cdot 16 + 15)_{10} \\
&= (768 + 192 + 15)_{10} \\
&= 975_{10}
\end{aligned}
$$

[3] The Logic of Quantified Statements

▼ [3.1] Predicates and Quantified Statements I

- A **predicate** is a sentence that contains a finite number of variables and becomes a statement when values are substituted into those variables
  - Thus, they are more generalized than previous statements
- The **domain** of a predicate variable is the set of all values that may be substituted in its place
- Let $P(x)$ be the predicate "$x^2 > x$" with domain the set $\mathbb{R}$. Write $P(2)$, $P(\frac{1}{2})$, and $P(-\frac{1}{2})$ and indicate which statements are true and which are false
  - $P(2)$: $2^2 > 2$ or $4 > 2$: **True**
  - $P(\frac{1}{2})$: $(\frac{1}{2})^2 > \frac{1}{2}$ or $\frac{1}{4} > \frac{1}{2}$: **False**
  - $P(-\frac{1}{2})$: $(-\frac{1}{2})^2 > -\frac{1}{2}$ or $\frac{1}{4} > -\frac{1}{2}$: **True**
- Given predicate $P(x)$ where x has domain D, the **truth set** of $P(x)$ is the set of all elements of D that result in $P(x)$ being true. This is denoted as

$$\{\, x \in D \mid P(x) \,\}$$

- Let $Q(n)$ be the predicate "n is a factor of 8." Find the truth set of $Q(n)$ if
  - The domain of n is $\mathbb{Z}^+$
    - $\boxed{\{\,1,\ 2,\ 4,\ 8\,\}}$
  - The domain of n is $\mathbb{Z}$
    - $\boxed{\{\,-8,\ -4,\ -2,\ -1,\ 1,\ 2,\ 4,\ 8\,\}}$

The Universal Quantifier $\forall$

- **Quantifiers** are words that refer to quantities like "some" or "all" and tell for how many elements a predicate is true
- They are another way to extract a statement from a predicate
- The **universal quantifier** denoted by $\forall$ can be read as "for every," "for each," "for any," "given any," or "for all"
- "Every human being is a mortal" or "All human beings are mortal" can be expressed as:
  - "$\forall$ human beings x, x is mortal" or "For every human being x, x is mortal"
  - Or, even more abstractly: "$\forall x \in H$, x is mortal"

- A **universal statement** is a statement in the form "$\forall \in D$, $Q(x)$," thus the statement is true only if $Q(x)$ is true for all x in D
- Any value of x for which $Q(x)$ is false is a **counterexample** to the universal statement
- Let $D = \{1,\ 2,\ 3,\ 4,\ 5\}$ and consider the statements:
  - $\forall x \in D$, $x^2 \geq x$
    - "For every x in D, x^2 is greater than or equal to x." This is obviously a true universal statement, but it can also be proven mathematically as seen below
    - Let $\forall x \in \mathbb{R}$, $x^2 \geq x$

$$1^2 \geq 1, \quad 2^2 \geq 2, \quad 3^2 \geq 3, \quad 4^2 \geq 4, \quad 5^2 \geq 5$$

  - $\forall x \in \mathbb{R}$, $x^2 \geq x$
    - "For every x in $\mathbb{R}$, x^2 is greater than or equal to x." Intuitively, it is easy to tell that this universal statement has **counterexamples**, and can be proven mathematically

$$\left(\frac{1}{2}\right)^2 = \frac{1}{4} \ngeq \frac{1}{2}$$

The Existential Quantifier $\exists$

- The **existential quantifier** denoted by $\exists$ can be read as "there is"
- "There is a student in Math 140."
  - "$\exists$ a person p such that p is a student in Math 140."
  - Or, more formally: "$\exists p \in P$ such that p is a student in Math 140 where P is the set of all people."
- An **existential statement** is a statement in the form "$\exists x \in D$ such that $Q(x)$," thus the statement is only true if $Q(x)$ is true for at least one x in D
- For notation, indicate domain between $\exists$ and the variable or immediately after the variable when "such that" is inserted before the predicate
- $\exists m \in \mathbb{Z}^+$, such that $m^2 = m$
  - "There is at least one positive integer m such that $m^2 = m$." This existential statement is true

$$1^2 = 1$$

- Given $E = \{5,\ 6,\ 7,\ 8\}$ and $\exists m \in E$ such that $m^2 = m$
    - This existential statement is false

$$5^2 \neq 5, \quad 6^2 \neq 6, \quad 7^2 \neq 7, \quad 8^2 \neq 8$$

Formal vs. Informal Language

- Formal statements are comprised of mathematical symbols that indicate relationships
- Informal statements are comprised of words that directly communicate relationships
- Converting formal statements to informal statements can make them easier to understand
- $\forall x \in \mathbb{R}$, $x^2 \geq 0$
    - "Every real number has a nonnegative square."
    - "All real numbers have nonnegative squares."
- $\forall x \in \mathbb{R}$, $x^2 \neq -1$
    - "Every real number has a square that does not equal -1."
    - "No real number has a square that equals -1."
- $\exists m \in \mathbb{Z}^+$ such that $m^2 = m$
    - "There is a positive integer whose square is equal to itself."
    - "Some positive integers equal their own squares"
- Additionally, quantifiers can also trail the rest of the sentence
- "For any integer n, $2n$ is even."
    - "$2n$ is even for any integer n."
- "There exists at least one real number x such that $x^2 \leq 0$."
    - "$x^2 \leq 0$ for some real number x."
    - "$x^2 \leq 0$ for at least one real number x."
- Informal statements can also be formalized
- "All triangles have three sides."
    - $\forall$ triangle t, t has three sides
    - $\forall t \in T$, t has three sides (where T is the set of all triangles)
- "No dogs have wings."
    - $\forall$ dog d, d does not have wings

    - $\forall d \in D$, d does not have wings (where D is the set of all dogs)
- "Some programs are structured."
    - $\exists$ a program p such that p is structured
    - $\exists p \in P$ such that p is structured (where P is the set of all programs)

Universal Conditional Statements

- The **universal conditional statement** is denoted by

$$\forall x, \text{ if } P(x) \text{ then } Q(x).$$

- $\forall x \in \mathbb{R}$, if $x > 2$ then $x^2 > 4$
    - "If a real number is greater than 2, then its square is greater than 4."
    - "The square of any real number greater than 2 is greater than 4."
- Using the form "$\forall$___, if ____, then ____," informal statements can be converted into formal ones
- "If a real number is an integer, then it is a rational number."
    - $\forall x \in \mathbb{R}$, if $x \in \mathbb{Z}$ then $x \in \mathbb{Q}$
- "All bytes have eight bits."
    - $\forall x$, if x is a byte, then x has eight bits
- "No fire trucks are green."
    - $\forall x$, if x is a fire truck, then x is not green
- As seen in some of the previous examples, universal conditional statements do not have to **explicitly** include the domain of variables

Equivalent Forms of Universal and Existential Statements

- A statement of the form $\forall x \in U$, if $P(x)$ then $Q(x)$ can be rewritten as $\forall x \in D$, $Q(x)$
    - This is done by narrowing U to be the subset D with all values of x that make $P(x)$ true
- Conversely, $\forall x \in D$, $Q(x)$ can be rewritten as $\forall x$, if x is in D then $Q(x)$
- "All squares are rectangles."
    - $\forall x$, if x is a square then x is a rectangle
    - $\forall$ square x, x is a rectangle

- "There is an integer that is both prime and even."
    - $\exists n$ such that $\text{Prime}(x) \wedge \text{Even}(x)$
    - $\exists$ prime number n such that $\text{Even}(x)$
    - $\exists$ even number n such that $\text{Prime}(x)$

Bound Variables and Scope

- A variable x is **bound** by the quantifier that controls it and its **scope** begins when the quantifier introduces it and ends at the end of the quantified statement
- In this example, two statements serve different purposes
    1. "For every integer x, $x^2 \geq 0$"
    2. "There exists a real number x such that $x^3 = 8$"
    - In this case, x is bound by the quantifier $\forall$ in statement 1

Implicit Quantification

- "If a number is an integer, then it is a rational number."
    - The statement above does not include any indicators of a universal statement, but it is equivalent to one
- **Implicit quantification** does not directly use a quantifier phrase, but the presence of something like the indefinite article "a" can be a clue
- Given that $P(x)$ and $Q(x)$ are two predicates with a common domain of D
    - $P(x) \implies Q(x)$ indicates that everything in the truth set of $P(x)$ is in the $Q(x)$ truth set, meaning $\forall x$, $P(x) \rightarrow Q(x)$
    - $P(x) \iff Q(x)$ indicates that $P(x)$ and $Q(x)$ have the same truth set, meaning $\forall x$, $P(x) \leftrightarrow Q(x)$
- For instance, given
    - $Q(n)$ means "n is a factor of 8"
    - $R(n)$ means "n is a factor of 4"
    - $S(n)$ means "$n < 5$ and $n \neq 3$"
    - Domain of n is $\mathbb{Z}^+$
    - **Relationship 1**
        - $\forall n$ in $\mathbb{Z}^+$, $R(n) \rightarrow Q(n)$

    - $\therefore R(n) \implies Q(n)$
    - Or "n is a factor of $4 \implies n$ is a factor of 8."
  - **Relationship 2**
    - $\forall n$ in $\mathbb{Z}^+$, $R(n) \leftrightarrow S(n)$
    - $\therefore R(n) \iff S(n)$
    - Or "n is a factor of 4 $\iff n < 5$ and $n \neq 3$"
    - However, since their truth sets are identical, $S(n)$ may have the same relationship with $Q(n)$
    - $\forall n$ in $\mathbb{Z}^+$, $S(n) \leftrightarrow Q(n)$
    - $\therefore S(n) \iff Q(n)$
    - Or "$n < 5$ and $n \neq 3 \iff n$ is a factor of 8"

Tarski's World

- **Tarski's World** is a program that provides pictures of blocks with different shapes, sizes, and colors each occupying their own cell
- The configuration can be described using logical operators
- There is also special notation to denote specific statements
  - $\text{Triangle}(x)$ means "x is a triangle"
  - $\text{Blue}(y)$ means "y is blue"
  - $\text{RightOf}(x, y)$ means "x is to the right of y (potentially a different row, however)"
- Objects' names are denoted by letters

▼ [3.2] Predicates and Quantified Statements II

Negations of Quantified Statements

- The negation of universal statement

$$\forall x \text{ in } D, Q(x)$$

- Is logically equivalent to the statement

$$\exists x \text{ in } D \text{ such that } \neg Q(x)$$

- Symbolically, this can be represented by

$$\neg\big(\forall x \in D,\, Q(x)\big) \equiv \exists x \in D \text{ such that } \neg Q(x)$$

- Thus, the negation of a universal statement “all are” is logically equivalent to an existential statement “some are not” or “there is at least one that is not”
- Quantified statements are logically equivalent when their truth sets are the same, even with different domains
- The negation of existential statement

$$\exists x \text{ in } D \text{ such that } Q(x)$$

- Is logically equivalent to the statement

$$\forall x \text{ in } D,\, \neg Q(x)$$

- Symbolically, this can be represented by

$$\neg\big(\exists x \text{ in } D \text{ such that } Q(x)\big) \equiv\ \forall x \in D,\, \neg Q(x)$$

- $\forall$ primes p, p is odd
    - $\exists$ a prime p such that p is not odd
- $\exists$ a triangle T such that the sum of the angles of T equals 200°
    - $\forall$ triangles T, the sum of the angles of T does not equal 200°
- When writing negations of informal statements, take the negation using the formal rule to avoid mistakes
- “No politicians are honest.”
    - **Formal version:** $\forall$ politicians x, x is not honest
    - **Formal negation:** $\exists$ a politician x such that x is honest
    - **Informal negation: “**Some politicians are honest.”
- “The number 1357 is not divisible by any integer between 1 and 37.”
    - **Formal version:** $\forall$ integer n between 1 and 37, 1357 is not divisible by n
    - **Formal negation:** $\exists$ integer n between 1 and 37 such that 1357 is divisible by n
    - **Informal negation:** ”The number 1357 is divisible by some integer between 1 and 37.”

Negations of Universal Conditional Statements

- Notably, the negation of an "if-then" statement is logically equivalent to an "and" statement

$$\neg\big(P(x) \implies Q(x)\big) \equiv P(x) \wedge \neg Q(x)$$

- The negation of a universal conditional statement is given by

$$\forall x,\ P(x) \to Q(x) \equiv \exists x \text{ such that } \neg\big(P(x) \to Q(x)\big)$$

- Thus, the negation of the "if-then" can be plugged into the negation of the universal conditional statement

$$\forall x,\ P(x) \to Q(x) \equiv \exists x \text{ such that } P(x) \wedge \neg Q(x)$$

- $\forall$ person p, if p is blond then p has blue eyes
  - **Formal negation:** $\exists$ person p such that p is blond and does not have blue eyes
- "If a computer program has more than 100,000 lines, then it contains a bug."
  - **Informal negation:** "There is at least one computer program that has more than 100,000 lines and does not contain a bug."

The Relation among $\forall$, $\exists$, $\wedge$, and $\vee$

- The negation of a "for all" statement is a "there exists" statement and vice versa
- This is analogous to De Morgan's laws, which highlight that an "and" statement can be negated into an "or" statement
- Thus, universal statements are generalizations of "and" statements, and existential statements are generalizations of "or" statements

$$\forall x \in D,\ Q(x) \equiv Q(x_1) \wedge Q(x_2) \wedge \cdots \wedge Q(x_n)$$

$$\exists x \in D \text{ such that } Q(x)\ \equiv Q(x_1) \vee Q(x_2) \vee \cdots \vee Q(x_n)$$

Vacuous Truth of Universal Statements

- A universal statement can be **vacuously true** or **true by default** if its negation is false
  - This is because a statement can only be true if its negation is false
  - Related to **vacuous truth** with "If, then" statements
  - Thus, the statement

$$\forall x \text{ in } D, \text{if } P(x) \text{ then } Q(x)$$

- Is vacuously true if, and only if, $P(x)$ is false for every x in D
- In the case of a bowl filled with balls colored gray or blue, any statements concerning the balls' colors is true by default if the bowl is empty

Variants of Universal Conditional Statements

- A conditional statement has a **contrapositive**, a **converse**, and an **inverse**

$$\forall x \in D, \text{if } P(x) \text{ then } Q(x)$$

$$\textbf{Contrapositive}$$
$$\forall x \in D, \text{if } \neg Q(x) \text{ then } \neg P(x)$$

$$\textbf{Converse}$$
$$\forall x \in D, \text{if } Q(x) \text{ then } P(x)$$

$$\textbf{Inverse}$$
$$\forall x \in D, \text{if } \neg P(x) \text{ then } \neg Q(x)$$

- "If a real number is greater than 2, then its square is greater than 4."
  - **Formal version:** $\forall x \in \mathbb{R}$, if $x > 2$ then $x^2 > 4$
  - **Formal contrapositive:** $\forall x \in \mathbb{R}$, if $x^2 \leq 4$ then $x \leq 2$
  - **Informal contrapositive: "**If a real number's square is less than or equal to 4, then it is less than ore equal to 2."
  - **Formal converse:** $\forall x \in \mathbb{R}$, if $x^2 \leq 4$ then $x > 2$
  - **Informal converse: "**If a real number's square is greater than 4, then it is greater than 2."
  - **Formal inverse:** $\forall x \in \mathbb{R}$, if $x \leq 2$ then $x^2 \leq 4$
  - **Informal inverse:** "If a real number is less than or equal to 2, then its square is less than or equal to 4."

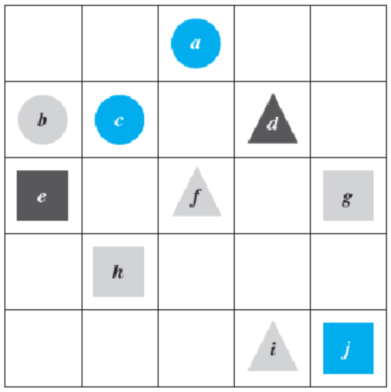
Necessary and Sufficient Conditions, Only If

- Necessary, sufficient, and only if conditions may be applied to universal conditional statements as well
- "$\forall x$, $r(x)$ is a **sufficient condition** for $s(x)$" means $\forall x$, if $r(x)$ then $s(x)$
- "$\forall x$, $r(x)$ is a **necessary condition** for $s(x)$" means $\forall x$, if $\neg r(x)$ then $\neg s(x)$

    - Applying contrapositive, $\forall x$, if $s(x)$ then $r(x)$
- "$\forall x$, $r(x)$ **only if** $s(x)$" means $\forall x$, if $\neg s(x)$ then $\neg r(x)$
    - Applying contrapositive, $\forall x$, if $r(x)$ then $s(x)$
    - Equivalent to the sufficient condition

▼ [3.3] Statements with Multiple Quantifiers

- When there are multiple quantifiers in a statement, the order of the quantifiers should communicate the order of the actions they suggest
- $\forall x$ in set D, $\exists y$ in set E such that x and y satisfy property $P(x, y)$
    - To prove this statement true, use any statement x in D and find a statement y in E such that the statements together satisfy $P(x, y)$
    - Different values of y can be found for different values of x



- "For every triangle x there is a square y such that x and y have the same color."
    - Given $x = d$, $y = e$ can be chosen (both are dark gray)
    - Given $x = f$ or $x = i$, $y = h$ or $y = g$ can be chosen (both are light gray)
- "There is a triangle x such that for every circle y, x is to the right of y"
    - Given $x = d$ or $x = i$, then given $y = a$ or $y = b$ or $y = c$, x is to the right of y

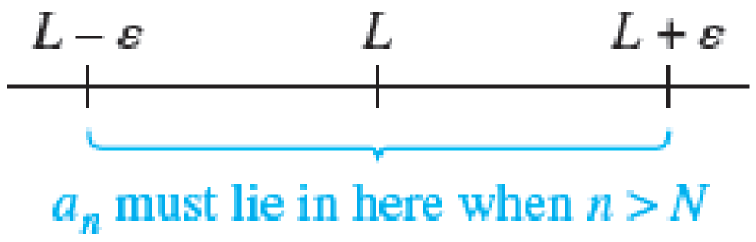
Translating from Informal to Formal Language

- Problems are generally stated in informal language, but are easier to solve in formal terms
- "The reciprocal of a real number a is a real number b such that $ab = 1$." The following are true statements
    - **Informal:** "Every nonzero number has a reciprocal."

- **Formal:** $\forall$ nonzero number u, $\exists$ a real number v such that $uv = 1$
- **Informal:** “There is a real number with no reciprocal.”
- **Formal:** $\exists$ real number u such that $\forall$ real number d, $cd \neq 1$

$$\lim_{n \to \infty} a_n = L$$

- “The limit of a sequence a_n as n goes to infinity equals L if, and only if, the values of a_n become arbitrarily close to L as n gets larger and larger without bound.”
  - $\forall \epsilon > 0$, $\exists$ an integer N such that $\forall$ integer n, if $n > N$ then $L - \epsilon < a_n < L + \epsilon$

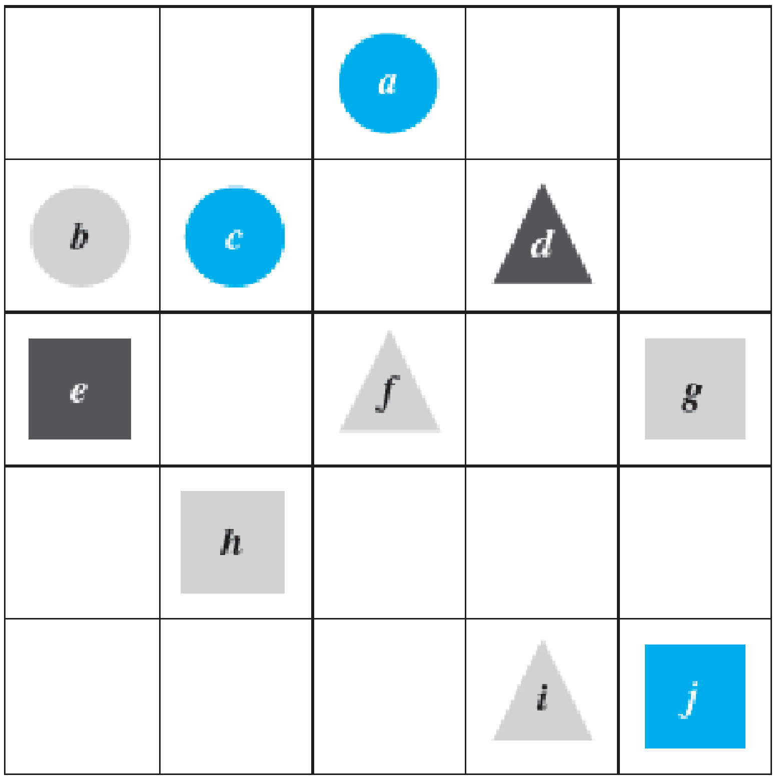


Ambiguous Language

- Informal language can be more ambiguous

Negations of Statements with More Than One Quantifier

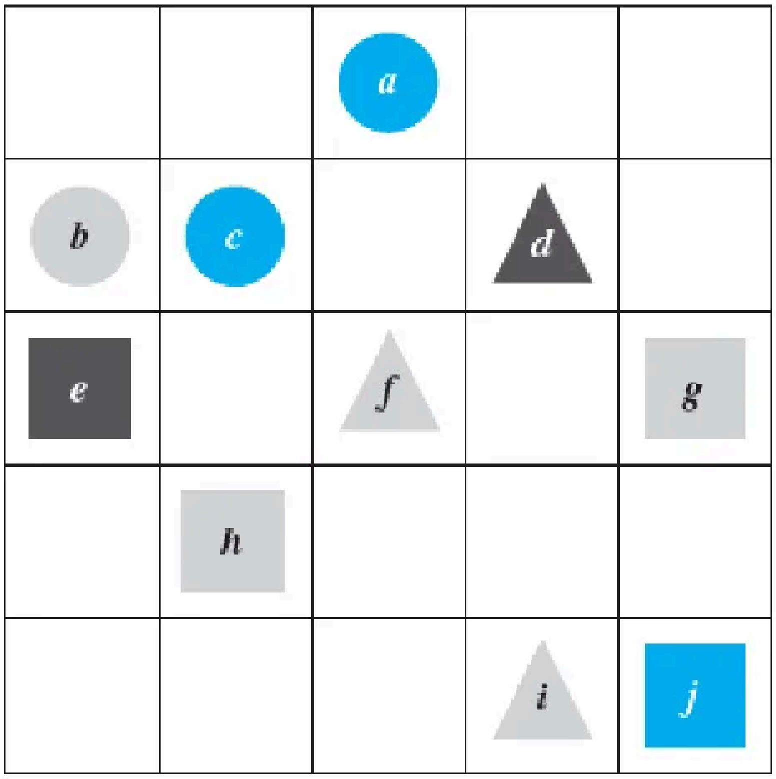
- Statements with more than one quantifier follow the same negation rules as statements with less
- The best strategy is to apply the negation one by one
- $\neg\big(\forall x$ in D, $\exists y$ in E such that $P(x, y)\big) \equiv \exists x$ in D such that $\neg\big(\exists y$ in E such that $P(x, y)\big) \equiv \exists x$ in D such that $\forall y$ in E, $\neg P(x, y)$



- "For every square x, there is a circle y such that x and y have the same color."
  - $\exists$ a square x such that $\neg\big(\exists$ a circle y such that x and y have the same color$\big)$
  - $\exists$ a square x such that $\forall$ circle y, x and y do not have the same color

Order of Quantifiers

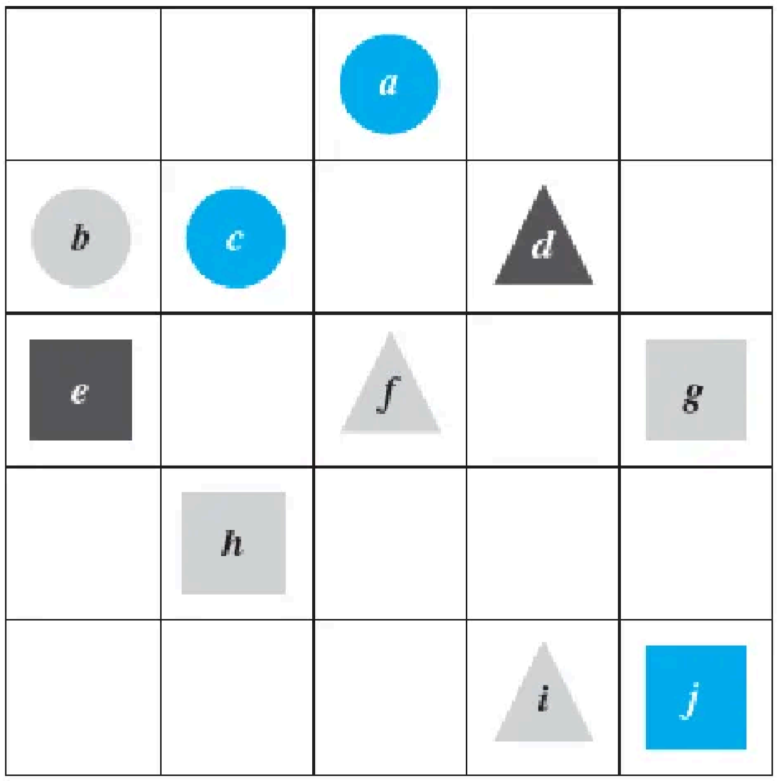
- In a statement containing both $\forall$ and $\exists$, the order of the quantifiers matters
- Conversely, if there is more than one quantifier of the same type, the order does not matter



- Due to the different order of the following statements, they lack the same truth values
    - “For every square x there is a triangle y such that x and y have different colors”
    - “There exists a triangle y such that for every square x, x and y have different colors”
    - The first statement is true, but the second one is false

Formal Logical Notation

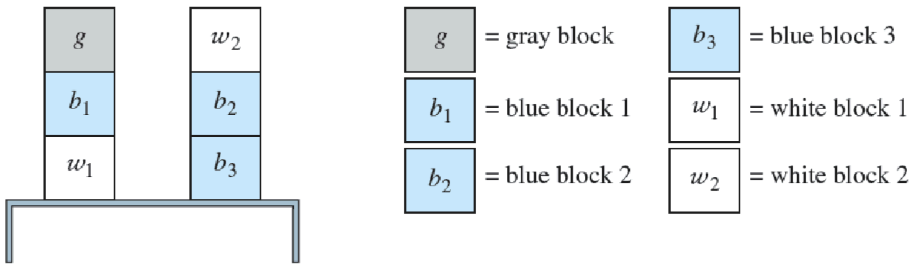
- In areas like computer science, logical statements are expressed in purely symbolic notation
- Even words like “such as” are omitted
- The two following facts are major parts of formalism
    - “$\forall x$ in D, $P(x)$” can be written as “$\forall x \left(x \text{ in } D \to P(x)\right)$”
    - “$\exists x$ in D such that $P(x)$” can be written as “$\exists x \left(x \text{ in } D \wedge P(x)\right)$”



- The main issue with formal logical notation is that it is not intuitive
- On the other hand, operations like taking negations can be replicated completely mechanically and programmed on computers
- **Language of first-order logic** refers to the symbols for quantifiers, variables, predicates, and logical connectives

Prolog

- **Prolog** programs are made up of a set of statements describing a situation with questions concerning the situation
- Prolog has built-in search and inference techniques to answer questions by deriving answers from given statements
    - Thus, the programmer does not need to write separate programs to answer specific questions



- $\text{isabove}(g, b_1)$
  - "g is above b_1"
- $\text{color}(b_1, \text{blue})$
  - "b_1 is colored blue"
- $\text{isabove}(X, Z)$ if $\text{isabove}(X, Y)$ and $\text{isabove}(Y, Z)$
  - "For all X, Y, and Z, if X is above Y and Y is above Z, then X is above Z"
- Additionally, there are corresponding questions for statements
- $?\text{isabove}(g, b_1)$ → **Yes**
- $?\text{isabove}(X, w_1)$ → $X = b_1$, $X = g$
  - This is equivalent to a predicate
  - For $X = g$, Prolog needs to infer the solution as it is not a given statement

▼ [3.4] Arguments with Quantified Statements

- The **rule of universal instantiation** states that if a property is true of everything in a set, then it is true of any particular thing in the set
- The following is one of the most well known examples of universal instantiation

$$\begin{gathered}\text{All men are mortal.}\\ \text{Socrates is a man.}\\ \therefore \text{Socrates is a mortal.}\end{gathered}$$

- Universal instantiation can also be applied to algebraic simplification problems

$$r^{k+1} \cdot r$$

- In this case, it can be used in tandem with two universal statements regarding exponents
  - For every real number x and for all integers m and n, $x^m \cdot x^n = x^{m+n}$

- For every real number x, $x = x^1$

$$\begin{aligned} r^{k+1} \cdot r &= r^{k+1} \cdot r^1 \\ &= r^{(k+1)+1} \\ &= r^{k+2} \end{aligned}$$

- In each step, universal instantiation is used
  - For the first step:
    1. For every real number x, $x^1 = x$ **(universal truth)**
    2. r is a particular real number **(particular instance)**
    3. $\therefore r^1 = r$
  - For the second step:
    1. For every real number x and for all real integers m and n, $x^m \cdot x^n = x^{m+n}$ **(universal truth)**
    2. r is a particular real number and $k+1$ and 1 are particular integers **(particular instance)**
    3. $\therefore r^{k+1} \cdot r^1 = r^{(k+1)+1}$

Universal Modus Ponens

- The rule of universal instantiation can be combined with modus ponens to get the valid form of the **universal modus ponens** argument form

$$\begin{gathered} \forall x, \text{if } P(x) \text{ then } Q(x). \\ P(a) \text{ for a particular } a. \\ \therefore Q(a). \end{gathered}$$

- Informally, if x makes $P(x)$ true, then x makes $Q(x)$ true. a makes $P(x)$ true. Therefore, a makes $Q(x)$ true
- This can be applied in a proof

Universal Modus Tollens

- As with universal modus ponens, universal modus tollens is an expansion of universal instantiation

$$\begin{gathered}\forall x, \text{if } P(x) \text{ then } Q(x).\\ \neg Q(a), \text{for a particular } a.\\ \therefore \neg P(a)\end{gathered}$$

- Informally, if x makes $P(x)$ true, then x makes $Q(x)$ true. a does not make $Q(x)$ true. Therefore, a does not make $P(x)$ true
- This can also be applied in a proof

$$\begin{gathered}\text{All humans are mortals.}\\ \text{Zeus is not a mortal.}\\ \therefore \text{Zeus is not a human.}\end{gathered}$$

- Imagining $H(x)$ as "x is human" and $M(x)$ as "x is mortal", this can be rewritten formally as $\forall x$, if $H(x)$ then $M(x)$
- If Z is Zeus, then the second line can be rewritten as $\neg M(Z)$ and the third line can be rewritten as $\neg H(Z)$, showing that the argument is valid by universal modus tollens

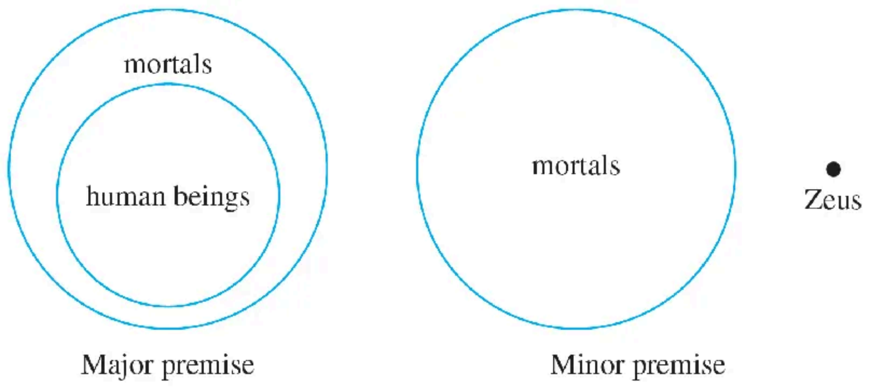
Proving Validity of Arguments with Quantified Statements

- An argument form is **valid** if, and only if, its form is valid and is **sound** if, and only if, both its form and its premises are true

Using a Diagram to Test For Validity

- Different sets can be pictured as disks of different sizes
  - Subsets would be disks inside other disks

$$\begin{gathered}\text{All humans are mortals.}\\ \text{Zeus is not a human.}\\ \therefore \text{Zeus is not a mortal.}\end{gathered}$$



- Since Zeus is outside the mortals disk, Zeus is necessarily outside the human beings disk. Thus, it is impossible for the premises of this argument to be true while the conclusion is false
    - The argument is valid
    - If the argument form has stated swapped the second and third lines, then Zeus would be outside of the human beings disc, but not necessarily outside of the mortals disk (invalid argument)
- Some arguments use the word “No”, and when using disks they may be represented by two disks that do not overlap
- The converse error also has a quantified form

$$\begin{aligned}&\forall x, \text{if } P(x) \text{ then } Q(x).\\ &Q(a) \text{ for a particular } a.\\ &\therefore P(a). \quad \text{(Invalid Conclusion)}\end{aligned}$$

- Informally, if x makes $P(x)$ true, then x makes $Q(x)$ true. a makes $Q(x)$ true. Therefore, a makes $P(x)$ true **(invalid conclusion)**
- Similarly, the inverse error has a quantified form

$$\begin{aligned}&\forall x, \text{if } P(x) \text{ then } Q(x).\\ &\neg P(a) \text{ for a particular } a.\\ &\therefore \neg Q(a). \quad \text{(Invalid Conclusion)}\end{aligned}$$

- Informally, if x makes $P(x)$ true, then x makes $Q(x)$ true. a does not make $P(x)$ true. Therefore, a does not make $P(x)$ true **(invalid conclusion)**

Creating Additional Forms of Argument

- Universal transitivity is the combination of transitivity and universal instantiation

$$\begin{gathered}\forall x\ P(x) \implies Q(x)\\ \forall x\ Q(x) \implies R(x)\\ \therefore \forall x\ P(x) \implies R(x)\end{gathered}$$

- Informally, any x that makes $P(x)$ true makes $Q(x)$ true. Any x that makes $Q(x)$ true makes $R(x)$ true. Therefore, any x that makes $P(x)$ true makes $R(x)$ true
- Often, universal transitivity may require finding equivalent forms and reordering premises in order to find the right path to the conclusion

Remark on the Converse and Inverse Errors

- Converse and inverse errors generally come from people misinterpreting major premises as biconditional statements rather than as condition statements
- “All the town criminals frequent the Den of Iniquity bar. John frequents the Den of Iniquity bar. Therefore, John is one of the town criminals.”
    - This is considered invalid due to a converse error, as while someone who is a criminal frequents the bar, that does not mean someone who frequents the bar is a criminal
    - As the major premise becomes more of a biconditional (e.g. “Hardly anyone but town criminals frequent the Den of Iniquity bar.”), then the argument becomes more valid
- Even then, a variation of the converse rule can be used to aid reasoning
    - If “for every x, if $P(x)$ then $Q(x)$” is a true statement and if “$Q(a)$ is true for a particular a”, then $P(a)$ has a *possibility* of being true