Discrete Mathematics

Contents

Discrete Mathematics

1. Logic

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

Logic

A formal system for describing knowledge and implementing reasoning on knowledge.

Logic consists of

1. A language describing knowledge (states of affairs) where its syntax describes how to make sentences and its semantics states how to interpret sentences
2. A set of rules for deducing the entailments of a set of sentences.

1-1. Propositional Logic

Propositional Logic

- *Propositional logic* treats simple sentences as atomic entities and constructs more complex sentences from simpler sentences using *Boolean connectives*.

Propositions and Proposition Variables

- *Definition*:
  1. A *proposition* is simply a declarative sentence *with a definite meaning*, having a *truth value* that's either *true* (T) or *false* (F) (never both, neither, or somewhere in between).
  2. A *proposition* (*statement*) may be denoted by a variable like *P, Q, R,…,* called a *proposition (statement) variable*.

- Note the difference between a proposition and a proposition variable.

Examples:

- “It is raining.” (In a given situation.)
- “Seoul is the capital of South Korea.”
- “1 + 2 = 3”

<u>But, the following are NOT propositions:</u>

- “Who’s there?” (interrogative, question)
- “La la la la la.” (meaningless interjection)
- “Just do it!” (imperative, command)
- “Yeah, I sorta dunno, whatever...” (vague)
- “1 + 2” (expression with a non-true/false value)

Operators / Connectives

1. *Operator* or *connective* combines one or more *operand* expressions into a larger expression (*e.g.*, "+" in numeric expressions).
2. *Unary* operators take 1 operand (*e.g.,* -3).
3. *binary* operators take 2 operands (*e.g.,* 3×4).
4. *Propositional* or *Boolean* operators operate on propositions or truth values instead of on numbers.

Some Popular Boolean Operators

Formal Name	Nickname	Arity	Symbol
Negation operator	NOT	Unary	$\neg$
Conjunction operator	AND	Binary	$\wedge$
Disjunction operator	OR	Binary	$\vee$
Exclusive-OR operator	XOR	Binary	$\oplus$
Implication operator	IMPLIES	Binary	$\rightarrow$
Biconditional operator	IFF	Binary	$\leftrightarrow$

Negation Operator

The unary *negation operator* "¬" (*NOT*) transforms a prop. into its logical *negation*.

E.g. If p = "I have brown hair."

then $\neg p$ = "I do not have brown hair."

Truth table for NOT:

p	$\neg p$
T	F
F	T

Operand column

Result column

Conjunction Operator

The binary *conjunction operator* "$\wedge$" (*AND*) combines two propositions to form their logical *conjunction*.

Example:

If p = "I will have salad for lunch." and q = "I will have steak for dinner.", then $p \wedge q$ = "I will have salad for lunch and I will have steak for dinner."

Conjunction Truth Table

- Note that a conjunction $p_1 \wedge p_2 \wedge \ldots \wedge p_n$ of n propositions will have 2^n rows in its truth table.

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

- Also: ¬ and ∧ operations together are sufficient to express *any* Boolean truth table!

Disjunction Operator

The binary *disjunction operator* "$\lor$" (*OR*) combines two propositions to form their logical *disjunction*.

p="My car has a bad engine."

q="My car has a bad carburetor."

$p \lor q$="Either my car has a bad engine, or my car has a bad carburetor."

Disjunction Truth Table

p	q	$p \lor q$
F	F	F
F	T	T
T	F	T
T	T	T

- Note that $p \lor q$ means that p is true, or q is true, or both are true!
- So, this operation is also called *inclusive or,* because it includes the possibility that both p and q are true.
- "$\neg$" and "$\lor$" together are also universal.

Nested Propositional Expressions

- Use parentheses to *group sub-expressions*:
  "I just saw my old friend, and either he's grown or I've shrunk." $= f \wedge (g \vee s)$
  - $(f \wedge g) \vee s$ would mean something different
  - $f \wedge g \vee s$ would be ambiguous

- By convention, "$\neg$" takes *precedence* over both "$\wedge$" and "$\vee$".
  - $\neg s \wedge f$ means $(\neg s) \wedge f$, not $\neg (s \wedge f)$

Example

Let p=“It rained last night”,
q=“The sprinklers came on last night,”
r=“The lawn was wet this morning.”

Translate each of the following into English:

$\neg p$ = “It didn’t rain last night.”

$r \wedge \neg p$ = “The lawn was wet this morning, and it didn’t rain last night.”

$\neg r \vee p \vee q$ = “Either the lawn wasn’t wet this morning, or it rained last night, or the sprinklers came on last night.”

Exclusive-Or Operator

The binary *exclusive-or operator* "$\oplus$" (*XOR*) combines two propositions to form their logical "exclusive or".

p = "I will earn an A in this course,"

q = "I will drop this course,"

$p \oplus q$ = "I will either earn an A for this course, or I will drop it (but not both!)"

Exclusive-Or Truth Table

- Note that $p \oplus q$ means that p is true, or q is true, but not both!
- This operation is called *exclusive or,* because it excludes the possibility that both p and q are true.

p	q	$p \oplus q$
F	F	F
F	T	T
T	F	T
T	T	F

Implication Operator

The *implication* $\overbrace{p}^{\text{antecedent}} \rightarrow \overbrace{q}^{\text{consequent}}$ states that p implies q.

I.e., If p is true, then q is true; but if p is not true, then q could be either true or false.

Example:

Let p = "You study hard."
q = "You will get a good grade."
$p \rightarrow q$ = "If you study hard, then you will get a good grade." (else, it could go either way)

Implication Truth Table

- $p \rightarrow q$ is false only when p is true but q is not true.
- $p \rightarrow q$ does not say that p causes q!
- $p \rightarrow q$ does not require that p or q are ever true!
- Example: "(1=0) $\rightarrow$ pigs can fly" is TRUE!

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

Examples

- "If this lecture ends, then the sun will rise tomorrow."
  True or *False*?
- "If Tuesday is a day of the week, then I am a penguin."
  True or *False*?
- "If 1+1=6, then Bush is president."
  True or *False*?
- "If the moon is made of green cheese, then I am richer than Bill Gates." *True* or *False*?

English Phrases Meaning $p \to q$

- "p implies q"
- "if p, then q"
- "if p, q"
- "when p, q"
- "whenever p, q"
- "q if p"
- "q when p"
- "q whenever p"
- "p only if q"
- "p is sufficient for q"
- "q is necessary for p"
- "q follows from p"
- "q is implied by p"

Converse, Inverse, Contrapositive

Some terminology, for an implication $p \rightarrow q$:

- Its *converse* is: $q \rightarrow p$.
- Its *inverse* is: $\neg p \rightarrow \neg q$.
- Its *contrapositive*: $\neg q \rightarrow \neg p$.
- One of these three has the *same meaning* (same truth table) as $p \rightarrow q$. Can you figure out which?

How do we know for sure?

Proving the equivalence of $p \rightarrow q$ and its contrapositive using truth tables:

p		q	$\neg q$		$\neg p$	$p \rightarrow q$	$\neg q \rightarrow \neg p$
F	$\rightarrow$	F	T	$\rightarrow$	T	T	T
F	$\rightarrow$	T	F	$\rightarrow$	T	T	T
T	$\not\rightarrow$	F	T	$\not\rightarrow$	F	F	F
T	$\rightarrow$	T	F	$\rightarrow$	F	T	T

Biconditional operator

The *biconditional* $p \leftrightarrow q$ states that p is true *if and only if (iff)* q is true.

p = "You can take the flight."

q = "You buy a ticket"

$p \leftrightarrow q$ = "You can take the flight if and only if you buy a ticket."

Biconditional Truth Table

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

- $p \leftrightarrow q$ means that p and q have the same truth value.
- Note this truth table is the exact opposite of $\oplus$'s!
  - $p \leftrightarrow q$ means $\neg(p \oplus q)$
- $p \leftrightarrow q$ does not imply p and q are true, or cause each other.

Boolean Operations Summary

- We have seen 1 unary operator (out of the 4 possible) and 5 binary operators (out of the 16 possible). Their truth tables are below.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
F	F	T	F	F	F	T	T
F	T	T	F	T	T	T	F
T	F	F	F	T	T	F	F
T	T	F	T	T	F	T	T

Well-formed Formula (wff) for Propositional Logic

- *Definition*:
  1. Any statement variable is a wff.
  2. For any wff p, $\neg p$ is a wff.
  3. If p and q are wffs, then $(p \wedge q)$, $(p \vee q)$, $(p \rightarrow q)$ and $(p \leftrightarrow q)$ are wffs.
  4. A finite string of symbols is a wff only when it is constructed by steps 1, 2, and 3.

Examples

- By definition of a wff,
  - wff: $\neg(P \wedge Q)$, $(P \rightarrow (P \vee Q))$, $(\neg P \wedge Q)$,
    $((P \rightarrow Q) \wedge (Q \rightarrow R)) \leftrightarrow (P \rightarrow R))$,
  - not wff: $(P \rightarrow Q) \rightarrow (\wedge Q)$, $(P \rightarrow Q,$

Tautology

- *Definition*:

  A well-formed formula (wff) is a *tautology* if for every truth value assignment to the variables appearing in the formula, the formula has the value of true.

- Example: $(p \vee \neg p)$

Substitution Instance

- *Definition*:

  A wff *A* is a substitution instance of another formula *B* if *A* is formed from *B* by substituting formulas for variables in *B* under condition that the same formula is substituted for the same variable each time that variable is occurred.

- *Theorem*:

  A substitution instance of a tautology is a tautology

Contradiction

- *Definition*:

  A wff is a *contradiction* if for every truth value assignment to the variables in the formula, the formula has the value of false.

- Example: $(p \wedge \neg p)$

Valid Consequence

- *Definition*:

  A (well-formed) formula B is a *valid consequence* of a formula A, denoted by $A \models B$, if for all truth assignments to variables appearing in A and B, the formula B has the value of true whenever the formula A has the value of true.

- *Definition*:

  A formula B is a *valid consequence* of a formula $A_1, \ldots, A_n$ ($A_1, \ldots, A_n \models B$) if for all truth value assignments to the variables appearing in $A_1, \ldots, A_n$ and B, the formula B has the value of true whenever the formula $A_1, \ldots, A_n$ have the value of true.

- *Theorem*:

  $A \models B$ *if and only if* $\models (A \rightarrow B)$

- *Theorem*:

  $A_1, \ldots, A_n \models B$ *if and only if* $(A_1 \wedge \ldots \wedge A_n) \models B$

- *Theorem*:

  $A_1, \ldots, A_n \models B$ *if and only if*

  $(A_1 \wedge \ldots \wedge A_{n\text{-}1}) \models (A_n \rightarrow B)$

Logical Equivalence

- *Definition*:

  Two wffs, *A* and *B,* are logically equivalent

  if and only if *A* and *B* have the same truth values for every truth value assignment to all variables contained in *A* and *B*.

- *Theorem*:

  If a formula A is equivalent to a formula B then $\models A \leftrightarrow B$.

- *Theorem*:

  If a formula D is obtained from a formula A by replacing a part of A, say C, which is itself a formula, by another formula B such that $C \Leftrightarrow B$, then $A \Leftrightarrow D$

Proving Equivalence via Truth Tables

- Example: Prove that $p \vee q \Leftrightarrow \neg(\neg p \wedge \neg q)$.

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

Equivalence Theorems

- *Identity*: $p \wedge \mathrm{T} \Leftrightarrow p$ $\quad p \vee \mathrm{F} \Leftrightarrow p$
- *Domination*: $p \vee \mathrm{T} \Leftrightarrow \mathrm{T}$ $\quad p \wedge \mathrm{F} \Leftrightarrow \mathrm{F}$
- *Idempotent*: $p \vee p \Leftrightarrow p$ $\quad p \wedge p \Leftrightarrow p$
- *Double negation:* $\neg\neg p \Leftrightarrow p$
- *Commutative*: $p \vee q \Leftrightarrow q \vee p$ $\quad p \wedge q \Leftrightarrow q \wedge p$
- *Associative*: $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

  $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$

- *Distributive*: $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$

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

- *De Morgan's*:

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

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

- *Trivial tautology/contradiction*:

  $p \vee \neg p \Leftrightarrow \mathrm{T}$ $\qquad$ $p \wedge \neg p \Leftrightarrow \mathrm{F}$

Defining Operators via Equivalences

Using equivalences, we can *define* operators in terms of other operators.

- Exclusive or: $p \oplus q \Leftrightarrow (p \vee q) \wedge \neg(p \wedge q)$
  $p \oplus q \Leftrightarrow (p \wedge \neg q) \vee (q \wedge \neg p)$
- Implies: $p \rightarrow q \Leftrightarrow \neg p \vee q$
- Biconditional: $p \leftrightarrow q \Leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$
  $p \leftrightarrow q \Leftrightarrow \neg(p \oplus q)$

Examples

Let p and q be the proposition variables denoting

p: It is below freezing.

q: It is snowing.

Write the following propositions using variables, p and q, and logical connectives.

a) It is below freezing and snowing. $p \wedge q$

b) It is below freezing but not snowing. $p \wedge \neg q$

c) It is not below freezing and it is not snowing. $\neg p \wedge \neg q$

d) It is either snowing or below freezing (or both). $p \vee q$

e) If it is below freezing, it is also snowing. $p \rightarrow q$

f) It is either below freezing or it is snowing, but it is not snowing if it is below freezing. $(p \vee q) \wedge (p \rightarrow \neg q)$

g) That it is below freezing is necessary and sufficient for it to be snowing $p \leftrightarrow q$

1-2. Predicate Logic

(First-order) Predicate Logic

- *Predicate logic* represents a sentence in terms of objects and predicates on objects (i.e., properties of objects or relations between objects), as well as Boolean connectives and quantifiers.

- In *propositional logic* every expression is a sentence, which represents a fact. *First-order predicate logic* has sentences, but it also has *terms*, which represent objects. *Constant symbols, variables,* and *function symbols* are used to build *terms*, and *quantifiers* and *predicate symbols* are used to build sentences.

Syntax and Semantics

- Constant symbols: *A, B, John,…*
- Variables: *x, y, z, …*
- Predicate symbols: *ROUND, BROTHER,…* where a predicate symbol refers to a particular relation in the model. For example, the *BROTHER* symbol referring to the relation of brotherhood is a binary predicate symbol having two objects.
- Function symbols: *father, color,…* where a function symbol maps its objects into some object.

where predicate and function symbols are often given by mnemonic strings.

Terms

- A *term* is a logical expression that refers to an object, which is defined as follows:
- *Definition:*

  1. Constant symbols and variables are *terms*.

  2. If *x* is a *term* and *h* is a function symbol, *h(x)* is a term.

  3. A finite string is a term only when it is constructed by steps 1 and 2.

- Examples:

  x, John, color(x), father(John), mother(father(John))

Functions and Predicates

- Arguments of functions and predicates are given by *terms*.

- Examples:

father(John), mother(Sue), father(mother(Sue)),

MARRIED(John, Sue), FEMALE(x), MEMBER(Sue,y)

PARENT(mother(Sue), Tom)

Universe of Discourse (U.D.)

- *Definition*:

  The collection of values that a variable x can take is called x's *universe of discourse*.

Quantifiers

- *Definition:*
  1. *Quantifiers* provide a notation that allows us to *quantify* (count) *how many* objects in the universe of discourse satisfy a given predicate.
  2. "$\forall$" is the FOR ALL or *universal* quantifier.
     $\forall x\, P(x)$ means for all x in the u.d., P holds.
  3. "$\exists$" is the EXISTS or *existential* quantifier.
     $\exists x\, P(x)$ means there exists an x in the u.d. (that is, 1 or more) such that $P(x)$ is true.

Universal Quantifier ∀

- Example:
  Let the u.d. of *x* be parking spaces at SNU.
  Let $P(x)$ be the *predicate* "*x* is full."
  Then the *universal quantification of* $P(x)$, $\forall x\, P(x)$, is the *proposition:*
  1. "All parking spaces at SNU are full."
  2. "Every parking space at SNU is full."
  3. "For each parking space at SNU, that space is full."

Existential Quantifier $\exists$

- Example:
  Let the u.d. of x be parking spaces at SNU.
  Let $P(x)$ be the *predicate* "x is full."
  Then the *existential quantification of* $P(x)$, $\exists x\, P(x)$, is the *proposition*:
  1. "Some parking space at SNU is full."
  2. "There is a parking space at SNU that is full."
  3. "At least one parking space at SNU is full."

Free and Bound Variables

- *Definition*:
  1. An expression like $P(x)$ is said to have a *free variable* x (meaning, x is undefined).
  2. A quantifier (either $\forall$ or $\exists$) *operates* on an expression having one or more free variables, and *binds* one or more of those variables, to produce an expression having one or more *bound variables*.

Examples

1. $P(x,y)$ has 2 free variables, x and y.
2. $\forall x\ P(x,y)$ has 1 free variable y, and one bound variable x.
3. $\forall x\ \forall y\ P(x,y)$ has zero free variables, which represents a proposition.

Nesting of Quantifiers

Example:

Let the u.d. of x and y be people.

Let $L(x,y)$="x likes y"

(A predicate with 2 free variables).

Then $\exists y\ L(x,y)$ = "There is someone whom x likes."

(A predicate with 1 free variable, x)

Then $\forall x\ \exists y\ L(x,y)$ = "Every one has someone whom they like."

(A predicate with 0 free variables)

Well-formed Formula (wff) for Predicate Logic

- *Definition*:

  A wff for (the first-order) predicate logic

  1. Every predicate formula is a wff.
  2. If P is a wff, $\neg P$ is a wff.
  3. Two wffs parenthesized and connected by $\wedge$, $\vee$, $\leftrightarrow$, $\rightarrow$ form a wff.
  4. If P is a wff and x is a variable then $(\forall x)P$ and $(\exists x)P$ are wffs.
  5. A finite string of symbols is a wff only when it is constructed by steps 1-4.

Examples

Let $R(x,y)$="x relies upon y". Express the following in unambiguous English:

1. $\forall x\, \exists y\, R(x,y)$ = Everyone has *someone* to rely on.
2. $\exists y\, \forall x\, R(x,y)$ = There's a poor overburdened soul whom *everyone* relies upon (including himself)!
3. $\exists x\, \forall y\, R(x,y)$ = There's some needy person who relies upon *everybody* (including himself).
4. $\forall y\, \exists x\, R(x,y)$ = Everyone has *someone* who relies upon them.
5. $\forall x\, \forall y\, R(x,y)$ = *Everyone* relies upon *everybody*. (including themselves)!

Natural language is ambiguous!

- "Everybody likes somebody."
  - For everybody, there is somebody they like,
    - $\forall x\ \exists y\ Likes(x,y)$ [Probably more likely.]
  - or, there is somebody (a popular person) whom everyone likes.
    - $\exists y\ \forall x\ Likes(x,y)$
- "Somebody likes everybody."
  - Same problem: Depends on context, emphasis.

More to Know About Binding

- $\forall x\ \exists x\ P(x)$ - x is not a free variable in $\exists x\ P(x)$, therefore the $\forall x$ binding isn't used.
- $(\forall x\ P(x)) \wedge Q(x)$ - The variable x is outside of the *scope* of the $\forall x$ quantifier, and is therefore free. Not a proposition!
- $(\forall x\ P(x)) \wedge (\exists x\ Q(x))$ – This is legal, because there are 2 different x's!

Quantifier Equivalence Laws

- Definitions of quantifiers: If u.d.=$a,b,c,\ldots$
  $\forall x\, P(x) \Leftrightarrow P(a) \wedge P(b) \wedge P(c) \wedge \ldots$
  $\exists x\, P(x) \Leftrightarrow P(a) \vee P(b) \vee P(c) \vee \ldots$

- From those, we can prove the laws:
  $\forall x\, P(x) \Leftrightarrow \neg(\exists x\, \neg P(x))$
  $\exists x\, P(x) \Leftrightarrow \neg(\forall x\, \neg P(x))$

More Equivalence Laws

- $\forall x\ \forall y\ P(x,y) \Leftrightarrow \forall y\ \forall x\ P(x,y)$
  $\exists x\ \exists y\ P(x,y) \Leftrightarrow \exists y\ \exists x\ P(x,y)$

- $\forall x\ (P(x) \wedge Q(x)) \Leftrightarrow (\forall x\ P(x)) \wedge (\forall x\ Q(x))$
  $\exists x\ (P(x) \vee Q(x)) \Leftrightarrow (\exists x\ P(x)) \vee (\exists x\ Q(x))$

Defining New Quantifiers

- *Definition*:

  $\exists! x\, P(x)$ is defined to mean "$P(x)$ is true of *exactly one* x in the universe of discourse."

- Note that $\exists! x\, P(x) \Leftrightarrow \exists x\, \big(P(x) \wedge \neg\exists y\, (P(y) \wedge (y \neq x))\big)$
  "There is an x such that $P(x)$, where there is no y such that $P(y)$ and y is other than x."

Higher-order Logic

- First-order logic gets its name from the fact that one can quantify over objects (the first-order entities that actually exist in the world) but not over relations or functions on those objects. Higher-order logic allows us to quantify over relations and functions as well as over objects. For example, in higher-order logic we can say that two objects are equal if and only if all properties applied to them are equivalent. Or we could say that two functions are equal if and only if they have the same value for all arguments:
  1. $(\forall x)(\forall y)\ (x=y) \leftrightarrow (\forall P)(P(x) \leftrightarrow P(y))$
  2. $(\forall f)(\forall g)\ (f=g) \leftrightarrow (\forall x)(f(x)=g(x))$

Logic for Monotonic Reasoning and Nonmonotonic Reasoning

- A logic is monotonic if, when some new sentences are added to the knowledge base, all the sentences entailed by the original knowledge base are still entailed by the new larger knowledge base. Otherwise, it is nonmonotonic.

Examples

Let $F(x, y)$ be the statement "*x loves y*," where the universe of discourse for both x and y consists of all people in the world. Use quantifiers to express each of these statements.

a) Everybody loves Jerry. $(\forall x)\ F(x, Jerry)$
b) Everybody loves somebody. $(\forall x)(\exists y)\ F(x,y)$
c) There is somebody whom everybody loves. $(\exists y)\ (\forall x)\ F(x,y)$
d) Nobody loves everybody. $\neg\ (\exists\ x)(\forall y)\ F(x,y)$
e) There is somebody whom Lydia does not love. $(\exists x)\ \neg\ F(Lydia,x)$
f) There is somebody whom no one loves. $(\exists x)(\forall y)\ \neg F(x,y)$
g) There is exactly one person whom everybody loves. $(\exists! x)(\forall y)F(y,x)$
h) There are exactly two people whom Lynn loves.

$(\exists x)\ (\exists y)\ ((x \neq y) \wedge F(Lynn,x) \wedge F(\text{Lynn},y) \wedge (\forall z)\ (\ F(\text{Lynn},z) \rightarrow (z=x) \vee (z=y)\ )\ )$

i) Everyone loves himself or herself $(\forall x)\ F(x,x)$
j) There is someone who loves no one besides himself or herself.

$(\exists x)\ (\forall y)\ F(x,y) \leftrightarrow \mathrm{x=y})$

Exercise

1. Let p, q, and r be the proposition variables such that

   p : You have the flu.

   q : You miss the final examination

   r : You pass the course

Express each of the following formulas as an English sentence.

(a) $(p \rightarrow \neg r) \vee (q \rightarrow \neg r)$

(b) $(p \wedge q) \vee (\neg q \wedge r)$

2. Let p, q, and r be the proposition variables such that

p : You get an A on the final exam.

q : You do every exercise in this book

r : You get an A in this class

Write the following propositions using p, q, r, and logical connectives.

(a) You get an A on the final, but you don't do every exercise in this book; nevertheless, you get an A in this class.

(b) Getting an A on the final and doing every exercise in this book is sufficient for getting an A in this class.

3. Assume the domain of all people.

Let $J(x)$ stand for "x is a junior", $S(x)$ stand for "x is a senior", and $L(x, y)$ stand for "x likes y". Translate the following into well-formed formulas:

(a) All people like some juniors.

(b) Some people like all juniors.

(c) Only seniors like juniors.

4. Let $B(x)$ stand for "x is a boy", $G(x)$ stand for "x is a girl", and $T(x,y)$ stand for "x is taller than y". Complete the well-formed formula representing the given statement by filling out ? part.

(a) Only girls are taller than than boys: $(?)(\forall y)((? \wedge T(x,y)) \rightarrow ?)$

(b) Some girls are taller than boys: $(\exists x)(?)(G(x) \wedge (? \rightarrow ?))$

(c) Girls are taller than boys only: $(?)(\forall y)((G(x) \wedge ?) \rightarrow ?)$

(d) Some girls are not taller than any boy: $(\exists x)(?)(G(x) \wedge (? \rightarrow ?))$

(e) No girl is taller than any boy: $(?)(\forall y)((B(y) \wedge ?) \rightarrow ?)$

1-3. Proofs and Inference Rules

Proof Terminology

- *Theorem*

  A statement that has been proven to be true.
- *Axioms*, *postulates*, *hypotheses*, *premises*

  Assumptions (often unproven) defining the structures about which we are reasoning.
- *Lemma*

  A minor theorem used as a stepping-stone to proving a major theorem.

- *Corollary*

  A minor theorem proved as an easy consequence of a major theorem.
- *Theory*

  The set of all theorems that can be proven from a given set of axioms.
- *Rules of inference*

  Patterns of deriving conclusions from hypotheses: *Sound* and *Complete*.

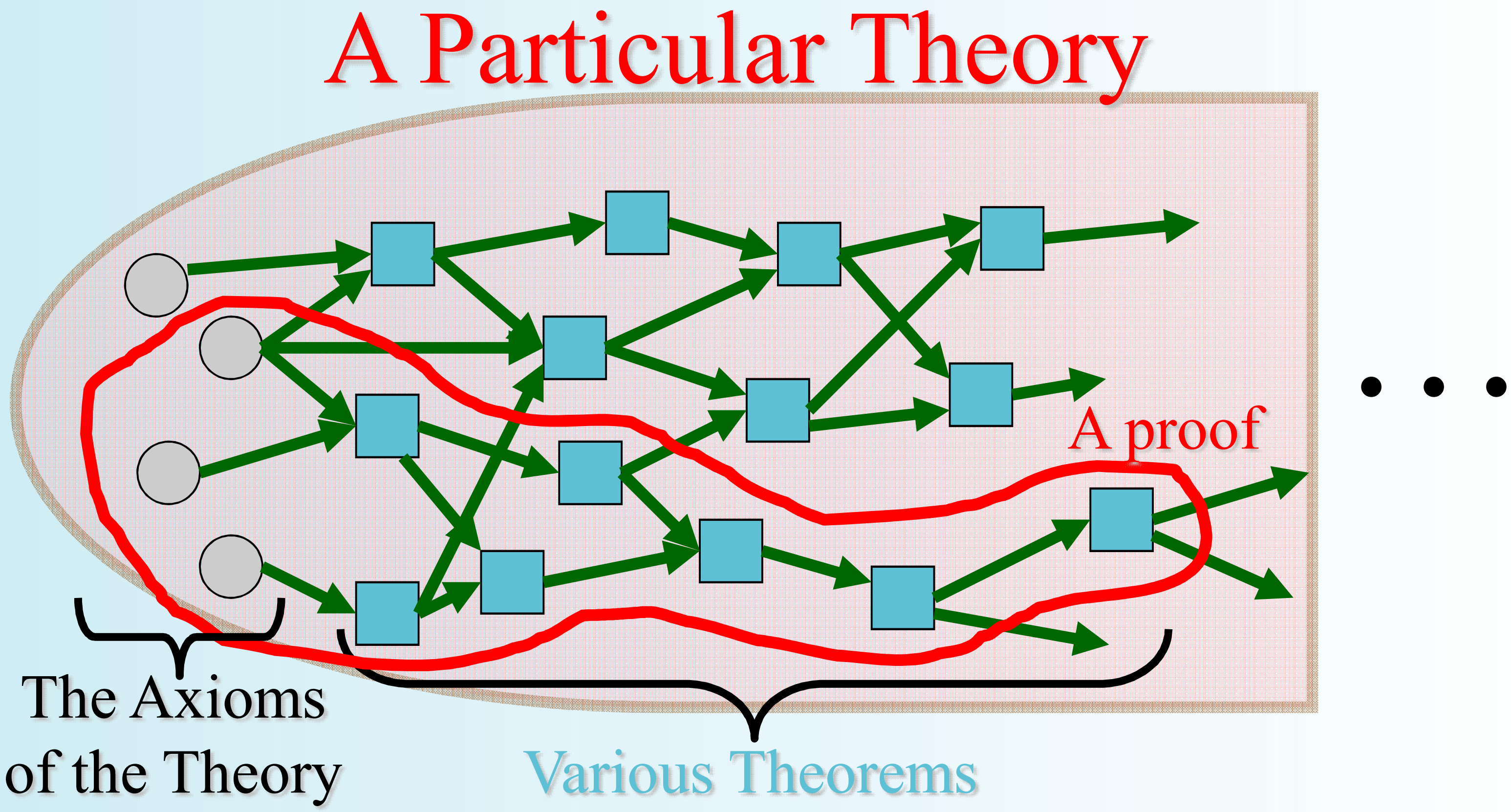
Depending on Inference Rules

- Deduction: $A \rightarrow B,\ A \ \Rightarrow\ B$
- Induction:

$$x \rightarrow B,\ y \rightarrow B,\ \ x,y \in A \ \Rightarrow\ \forall z \in A,\ \ z \rightarrow B$$

- Abduction: $A \rightarrow B,\ B \ \Rightarrow\ A$

Graphical Visualization



Inference Rules: General Form

- *Inference Rule*:

  Pattern establishing that if we know that a set of *antecedent* statements of certain forms are all true, then a certain related *consequent* statement is true (valid arguments).

- *antecedent 1*
  antecedent 2 …
  ―――――――――――
  ∴ *consequent*

  "∴" means "therefore"

Inference Rules: Implications

- Each logical inference rule corresponds to an implication that is a *tautology*.

- $$\boxed{\begin{array}{l} \textit{antecedent 1} \\ \underline{\textit{antecedent 2} \ \ldots} \\ \therefore \ \textit{consequent} \end{array}}$$ Inference rule

- Corresponding tautology:

  $((\textit{ante. 1}) \wedge (\textit{ante. 2}) \wedge \ldots) \Rightarrow \textit{consequent}$

Implication Tautologies

$I_1 \quad P \wedge Q \Rightarrow P$

$I_2 \quad P \wedge Q \Rightarrow Q$

$I_3 \quad P \Rightarrow P \vee Q$

$I_4 \quad Q \Rightarrow P \vee Q$

$I_5 \quad \neg P \Rightarrow P \rightarrow Q$

$I_6 \quad Q \Rightarrow P \rightarrow Q$

$I_7 \quad \neg(P \rightarrow Q) \Rightarrow P$

$I_8 \quad \neg(P \rightarrow Q) \Rightarrow \neg Q$

$I_9 \quad P, Q \Rightarrow P \wedge Q$

$I_{10} \quad \neg P, P \vee Q \Rightarrow Q$

$I_{11} \quad P, P \rightarrow Q \Rightarrow Q$

$I_{12} \quad \neg Q, P \rightarrow Q \Rightarrow \neg P$

$I_{13} \quad P \rightarrow Q, Q \rightarrow R \Rightarrow P \rightarrow R$

$I_{14} \quad P \vee Q, P \rightarrow R, Q \rightarrow R \Rightarrow R$

$I_{15} \quad (\forall x)A(x) \vee (\forall x)B(x)$

$\Rightarrow (\forall x)(A(x) \vee B(x))$

$I_{16} \quad (\exists x)(A(x) \wedge B(x))$

$\Rightarrow (\exists x)A(x) \wedge (\exists x)B(x)$

Biconditional Tautologies: Equivalences

E_1 $\neg\neg P \Leftrightarrow P$
E_2 $P \wedge Q \Leftrightarrow Q \wedge P$
E_3 $P \vee Q \Leftrightarrow Q \vee P$
E_4 $(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$
E_5 $(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$
E_6 $P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$
E_7 $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$
E_8 $\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$
E_9 $\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$
E_{10} $P \vee P \Leftrightarrow P$
E_{11} $P \wedge P \Leftrightarrow P$
E_{12} $R \vee (P \wedge \neg P) \Leftrightarrow R$
E_{13} $R \wedge (P \vee \neg P) \Leftrightarrow R$
E_{14} $R \vee (P \vee \neg P) \Leftrightarrow T$
E_{15} $R \wedge (P \wedge \neg P) \Leftrightarrow F$
E_{16} $P \rightarrow Q \Leftrightarrow \neg P \vee Q$
E_{17} $\neg(P \rightarrow Q) \Leftrightarrow P \wedge \neg Q$
E_{18} $P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P$
E_{19} $P \rightarrow (Q \rightarrow R) \Leftrightarrow (P \wedge Q) \rightarrow R$
E_{20} $\neg(P \leftrightarrow Q) \Leftrightarrow (P \leftrightarrow \neg Q)$
E_{21} $(P \leftrightarrow Q) \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$
E_{22} $(P \leftrightarrow Q) \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$
E_{23} $(\exists x)(\mathrm{A}(x) \vee B(x)) \Leftrightarrow (\exists x)A(x) \vee (\exists x)B(x)$
E_{24} $(\forall x)(\mathrm{A}(x) \wedge B(x)) \Leftrightarrow (\forall x)A(x) \wedge (\forall x)B(x)$
E_{25} $\neg(\exists x)A(x) \Leftrightarrow (\forall x)\neg A(x)$
E_{26} $\neg(\forall x)A(x) \Leftrightarrow (\exists x)\neg A(x)$
E_{27} $(\forall x)(A \vee B(x)) \Leftrightarrow A \vee (\forall x)B(x)$
E_{28} $(\exists x)(A \wedge B(x)) \Leftrightarrow A \wedge (\exists x)B(x)$
E_{29} $(\forall x)A(x) \rightarrow B \Leftrightarrow (\exists x)(A(x) \rightarrow B)$
E_{30} $(\exists x)A(x) \rightarrow B \Leftrightarrow (\forall x)(A(x) \rightarrow B)$
E_{31} $A \rightarrow (\forall x)B(x) \Leftrightarrow (\forall x)(A \rightarrow B(x))$
E_{32} $A \rightarrow (\exists x)B(x) \Leftrightarrow (\exists x)(A \rightarrow B(x))$
E_{33} $(\exists x)(A(x) \rightarrow B(x)) \Leftrightarrow (\forall x)A(x) \rightarrow \exists x B(x))$

Formal Proofs

- *Definition*:
  1. A formal proof of a conclusion C, given premises p_1, $p_2, \ldots, p_n$ consists of a sequence of *steps*, each of which applies some inference rule to premises or to previously-proven statements (as antecedents) to yield a new true statement (the consequent).
  2. Inference Rules
     - Rule ***P*** : premise
     - Rule ***T*** : tautology
     - Rule ***CP*** : conditional premise

- Note that a proof demonstrates that *if* the premises are true, *then* the conclusion is true.

Examples:

1. Suppose we have the following premises:
   (1) It is not sunny and it is cold.
   (2) We will swim only if it is sunny.
   (3) If we do not swim, then we will canoe.
   (4) If we canoe, then we will be home early.

Given these premises, prove using inference rules the theorem, “We will be home early”.

Proof:

Let us adopt the following abbreviations:

sunny = "It is sunny"; *cold* = "It is cold";

swim = "*W*e will swim"; *canoe* = "We will canoe";

early = "We will be home early".

Then, the premises can be represented by the following formulas:

$\neg sunny \wedge cold$, $swim \rightarrow sunny$, $\neg swim \rightarrow canoe$, $canoe \rightarrow early$.

Based on these formulas, the *proof* would be

Step	Inference Rule
(1) $\neg sunny \wedge cold$	$\boldsymbol{P}$
(2) $\neg sunny$	$\boldsymbol{T}$, (1) and $\boldsymbol{I_1}$
(3) $swim \rightarrow sunny$	$\boldsymbol{P}$
(4) $\neg swim$	$\boldsymbol{T}$, (2), (3) and $\boldsymbol{I_{12}}$
(5) $\neg swim \rightarrow canoe$	$\boldsymbol{P}$
(6) $canoe$	$\boldsymbol{T}$, (4), (5) and $\boldsymbol{I_{11}}$
(7) $canoe \rightarrow early$	$\boldsymbol{P}$
(8) $early$	$\boldsymbol{T}$, (6), (7), and $\boldsymbol{I_{11}}$

2. Show that $(R \to S)$ can be derived from $(P \to (Q \to S))$, $(\neg R \vee P)$, and Q. (Instead of deriving $R \to S$ directly, we shall include R as an additional premise and show S can be derive from there premises.)

Proof:

Step		Inference Rule
(1)	$\neg R \vee P$	***P***
(2)	R	***P*** (assumed premise)
(3)	P	***T***, (1), (2) and $\boldsymbol{I_{10}}$
(4)	$P \to (Q \to S)$	***P***
(5)	$Q \to S$	***T***, (3), (4) and $\boldsymbol{I_{11}}$
(6)	Q	***P***
(7)	S	***T***, (5), (6) and $\boldsymbol{I_{11}}$
(8)	$R \to S$	***CP***, (2), (7)

3. Show that $S \vee R$ can be derived from $(P \vee Q)$, $(P \rightarrow R)$ and $(Q \rightarrow S)$.

Proof:

Step		Inference Rule
(1)	$P \vee Q$	$\boldsymbol{P}$
(2)	$\neg P \rightarrow Q$	$\boldsymbol{T}$, (1) , $\boldsymbol{E_1}$ and $\boldsymbol{E_{16}}$
(3)	$Q \rightarrow S$	$\boldsymbol{P}$
(4)	$\neg P \rightarrow S$	$\boldsymbol{T}$, (2), (3), and $\boldsymbol{I_{13}}$
(5)	$\neg S \rightarrow P$	$\boldsymbol{T}$, (4) , $\boldsymbol{E_{18}}$ and $\boldsymbol{E_1}$
(6)	$P \rightarrow R$	$\boldsymbol{P}$
(7)	$\neg S \rightarrow R$	$\boldsymbol{T}$, (5), (6), and $\boldsymbol{I_{13}}$
(8)	$S \vee R$	$\boldsymbol{T}$, (7), $\boldsymbol{E_{16}}$ and $\boldsymbol{E_1}$

Inference Rules for Quantifiers

- $\dfrac{\forall x\, P(x)}{\therefore P(o)}$ Universal Specification (***US***) (substitute *any* object o)
- $\dfrac{P(g)}{\therefore \forall x\, P(x)}$ (for *general* element g of u.d.) Universal Generalization (***UG***)
- $\dfrac{\exists x\, P(x)}{\therefore P(c)}$ Existential Specification (***ES***) (substitute *some* object c)
- $\dfrac{P(o)}{\therefore \exists x\, P(x)}$ (for some extant object o) Existential Generalization (***EG***)

Examples:

1. Show that

$(\forall x)\,(P(x) \rightarrow Q(x)) \wedge (\forall x)\,(Q(x) \rightarrow R(x)) \Rightarrow (\forall x)\,(P(x) \rightarrow R(x))$

Proof:

<u>Step</u>		<u>Inference Rule</u>
(1)	$(\forall x)\,(P(x) \rightarrow Q(x))$	***P***
(2)	$P(y) \rightarrow Q(y)$	***US***, (1)
(3)	$(\forall x)\,(Q(x) \rightarrow R(x))$	***P***
(4)	$Q(y) \rightarrow R(y)$	***US***, (3)
(5)	$P(y) \rightarrow R(y)$	***T***, (2), (4) and $\boldsymbol{I_{13}}$
(6)	$(\forall x)\,(P(x) \rightarrow R(x))$	***UG***, (5)

2. Show that from $(\exists x)\,(F(x) \wedge S(x)) \rightarrow (\forall y)\,(M(y) \rightarrow W(y))$ and $(\exists y)\,(M(y) \wedge \neg W(y))$, the conclusion $(\forall x)\,(F(x) \rightarrow \neg\, S(x))$ logically follows.

Proof:

Step		Inference Rule
(1)	$(\exists y)\,(M(y) \wedge \neg W(y))$	***P***
(2)	$M(z) \wedge \neg W(z)$	***ES***, (1)
(3)	$\neg\,(M(z) \rightarrow W(z))$	***T***, (2) and $\boldsymbol{E_{17}}$
(4)	$(\exists y) \neg\,(M(y) \rightarrow W(y))$	***EG***, (3)
(5)	$\neg(\forall y)(M(y) \rightarrow W(y))$	***T***, (4) and $\boldsymbol{E_{26}}$
(6)	$(\exists x)\,(F(x) \wedge S(x)) \rightarrow (\forall y)\,(M(y) \rightarrow W(y))$	***P***
(7)	$\neg(\exists x)\,(F(x) \wedge S(x))$	***T***, (5), (6) and $\boldsymbol{I_{12}}$
(8)	$(\forall x) \neg(F(x) \wedge S(x))$	***T***, (7) and $\boldsymbol{E_{25}}$
(9)	$\neg(F(x) \wedge S(x))$	***US***, (8)
(10)	$F(x) \rightarrow \neg\, S(x)$	***T***, (9), $\boldsymbol{E_8}$ and $\boldsymbol{E_{16}}$
(11)	$(\forall x)\,(F(x) \rightarrow \neg\, S(x))$	***UG***, (10)

Restriction

- ***UG*** applicable variable should not be free in any of the given premises
- ***UG*** should not be applied to the free variables after ***ES*** making some other variable free in a prior step.

$(\forall x)(\exists z)\ A(z,x)$	
$\Rightarrow (\exists z)A(z,x)$	*by* ***US***
$\Rightarrow A(z,x)$	*by* ***ES***
$\Rightarrow (\forall x)A(z,x)$	*by* ***UG*** (not allowed!)
$\Rightarrow (\exists z)\ (\forall x)A(z,x)$	*by* ***EG*** contradiction!

Proof Methods for Implications

For proving implications $p \rightarrow q$, we have:

- *Direct* proof: Assume p is true, and prove q.
- *Indirect* proof: Assume $\neg q$, and prove $\neg p$.
- *Vacuous* proof: Prove $\neg p$ by itself.
- *Trivial* proof: Prove q by itself.
- Proof by cases:
  Show $p \rightarrow (a \vee b)$, and $(a \rightarrow q)$ and $(b \rightarrow q)$.

Example of Direct Proof

- *Definition*:

  An integer n is called *odd* iff $n=2k+1$ for some integer k; n is *even* iff $n=2k$ for some k.

- *Axiom*:

  Every integer is either odd or even.

- *Theorem*:

  (For all numbers n) If n is an odd integer, then n^2 is an odd integer.

  Proof:

  If n is odd, then $n = 2k+1$ for some integer k. Thus, $n^2 = (2k+1)^2 = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1$. Therefore n^2 is of the form $2j + 1$ (with j the integer $2k^2 + 2k$), thus n^2 is odd. □

Example of Indirect Proof

- *Theorem*: (For all integers n)
  If $3n+2$ is odd, then n is odd.

  Proof:

  Suppose that the conclusion is false, *i.e.*, that n is even. Then $n=2k$ for some integer k. Then $3n+2 = 3(2k)+2 = 6k+2 = 2(3k+1)$. Thus $3n+2$ is even, because it equals $2j$ for integer $j = 3k+1$. So $3n+2$ is not odd. We have shown that $\neg(n \text{ is odd}) \rightarrow \neg(3n+2 \text{ is odd})$, thus its contra-positive $(3n+2 \text{ is odd}) \rightarrow (n \text{ is odd})$ is also true. □

Example of Vacuous Proof

- *Theorem*: If n is both odd and even, then $n^2 = n + n$.

  Proof:

  The statement "n is both odd and even" is necessarily false, since no number can be both odd and even. So, the theorem is vacuously true. □

Example of Trivial Proof

- *Theorem*: (For integers n) If n is the sum of two prime numbers, then either n is odd or n is even.

  Proof:

  Any integer n is either odd or even. So the conclusion of the implication is true regardless of the truth of the antecedent.

  Thus the implication is true trivially. □

Proof by Contradiction

1. A method for proving p.
2. Assume $\neg p$, and prove both q and $\neg q$ for some proposition q.
3. Thus $\neg p \rightarrow (q \wedge \neg q)$
4. $(q \wedge \neg q)$ is a trivial contradiction, equal to F
5. Thus $\neg p \rightarrow$F, which is only true if $\neg p$=F
6. Thus p is true.

Proving Existentials

1. A proof of a statement of the form $\exists x\, P(x)$ is called an *existence proof*.
2. If the proof demonstrates how to actually find or construct a specific element a such that $P(a)$ is true, then it is a *constructive* proof.
3. Otherwise, it is *nonconstructive*.

Constructive Existence Proof

- *Theorem*:

  There exists a positive integer n that is the sum of two perfect cubes in two different ways:

  - equal to $j^3 + k^3$ and $l^3 + m^3$ where j, k, l, m are positive integers, and $\{j,k\} \neq \{l,m\}$

  Proof:

  Consider $n = 1729$, $j = 9$, $k = 10$, $l = 1$, $m = 12$. Now just check that the equalities hold.

Nonconstructive Existence Proof

- *Theorem*:

  There are infinitely many prime numbers.

  Proof:

  Any finite set of numbers must contain a maximal element, so we can prove the theorem if we can just show that there is *no* largest prime number.

  I.e., show that for any prime number, there is a larger number that is *also* prime.

  More generally: For *any* number, $\exists$ a larger prime.

  Formally: Show $\forall n\ \exists p\ ((p>n) \rightarrow (p \text{ is prime}))$.

Given $n>0$, prove there is a prime $p>n$.

Consider $x = n!+1$. Since $x>1$, we know

$(x \text{ is prime}) \lor (x \text{ is composite})$.

Case 1: x is prime.

Obviously $x>n$, so let $p=x$ and we're done.

Case 2: x has a prime factor p.

But if $p \le n$, then $x \bmod p = 1$.

So $p>n$, and we're done.

Uniqueness Proof

- Some theorems assert the existence of a unique element with a particular property.

- To prove a statements of this type, we show following two parts.
  1. Existence: element x with a desired property exists
  2. Uniqueness: if $y \neq x$, then y does not have the desired property

Example of Uniqueness Proof

- *Theorem*:
  "Every integer has a unique additive inverse."

  Proof:

  If p is an integer, we find that $p+q=0$ where $p=-q$ and q is also an integer. Consequently, there exists an integer q such that $p+q=0$. (Existence)

  if r is an integer with $r \neq q$ such that $p+r=0$. then $p+q=p+r$. So We can show $q=r$, which contradicts our assumption $r \neq q$. Consequently, there is a unique integer q such that $p+q=0$. □

Exercise

1. Prove that the square of an even number is an even number using
   (a) A direct proof
   (b) An indirect proof
   (c) A proof by contradiction

2. Prove formally using inference rules that $R \wedge (P \vee Q)$ logically follows from $(P \vee Q)$, $(Q \to R)$, $(P \to M)$, and $\neg M$.

3. Prove that if n is a positive integer, then n is a even if and only if $7n+4$ is even.

4. Let P, Q, R and S be statement variables.

   Prove formally the following.

   (a) $\neg P \wedge Q,\ \neg Q \vee R,\ R \rightarrow S \Rightarrow P \rightarrow S$

   (b) $\neg P \wedge (P \vee Q) \Rightarrow Q$

5. Show the following implication.

   (a) $(\forall x)(P(x) \vee Q(x)),\ (\forall x)\neg P(x) \Rightarrow (\exists x)Q(x)$

   (b) $\neg((\exists x)P(x) \wedge Q(a)) \Rightarrow (\exists x)P(x) \rightarrow \neg Q(a)$

Discrete Mathematics

2. Sets

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

Introduction to Set Theory

- A *set* is a new type of structure, representing an *unordered* collection of zero or more *distinct* (different) objects.
- Set theory deals with operations between, relations among, and statements about sets.

Naive Set Theory

- A set is any collection of objects (*elements*) that we can describe. (Basic premise)

- The naive set theory, however, leads to logical inconsistencies, known as *paradoxes*:

  Russell's paradox:

  1. *A set being a member of itself:* Possible from the case that the set of concepts is itself a concept, and hence this set is apparently a member of itself. The assertions ($x \notin x$) and ($x \in x$) are therefore predicates which can be used to define sets:
  2. *Define S to be* $S=\{x \mid x \notin x\}$.
  3. *Is S a member of itself?*

- Set theory is formulated to avoid *Russell's paradox*: Restrictions on the ways in which sets can be related, which imply that *no set is permitted to be a member of itself*. (Other *paradoxes* exist?)

Basic notations for Sets

- For sets, we'll use variables S, T, U, …
- We can denote a set S in writing by listing all of its elements in curly braces:
  - $\{a, b, c\}$ is the set of 3 objects denoted by a, b, and c.
- *Set builder notation*: For any predicate symbol P, $\{x | P(x)\}$ is *the set of all x such that P(x).* (or *the set of all x holding the property P.)*

Basic properties of Sets

- Sets are inherently *unordered*:
  - No matter what objects a, b, and c denote, $\{a, b, c\} = \{a, c, b\} = \{b, a, c\} = \{b, c, a\} = \{c, a, b\} = \{c, b, a\}$.

- All elements are *distinct* (unequal); multiple listings make no difference!
  - If $a=b$, then $\{a, b, c\} = \{a, c\} = \{b, c\} = \{a, a, b, a, b, c, c, c, c\}$.
  - This set contains at most 2 elements!

Infinite Sets

- Conceptually, sets may be *infinite* (*i.e.,* not *finite*, without end, unending).

- Symbols for some special infinite sets:
  N = {1, 2, …} The **N**atural numbers.
  Z = {…, -2, -1, 0, 1, 2, …} The Zntegers.
  R = The "**R**eal" numbers, such as 374.1828471929498181917281943125…

- Infinite sets come in different sizes!

Empty Set

- *Definition*:

  A set which does not contain any elements is an empty set, denoted by $\varnothing$ or $\{\}$ or $\{x|\ \text{False}\}$

- Example:

  $x \notin \varnothing$ for any x

Subset and Superset

- *Definition*:

  Let S and T be any two sets. S is a subset of T *(T is a superset of S),* denoted by $S \subseteq T$, *if and only if* every element of S is an element of T, *i.e.,*

  $(\forall x)((x \in S) \rightarrow (x \in T))$.

- Example:

  $\varnothing \subseteq S$, $S \subseteq S$.

Set Equality

- *Definition*:

  Let A and B be any two sets. A and B are said to be equal *if and only if* they contain exactly the same elements, i.e., $A=B$ *if and only if* $(A \subseteq B) \wedge (B \subseteq A)$.

- Note that it does not matter *how the set is defined or denoted.*
- Example:

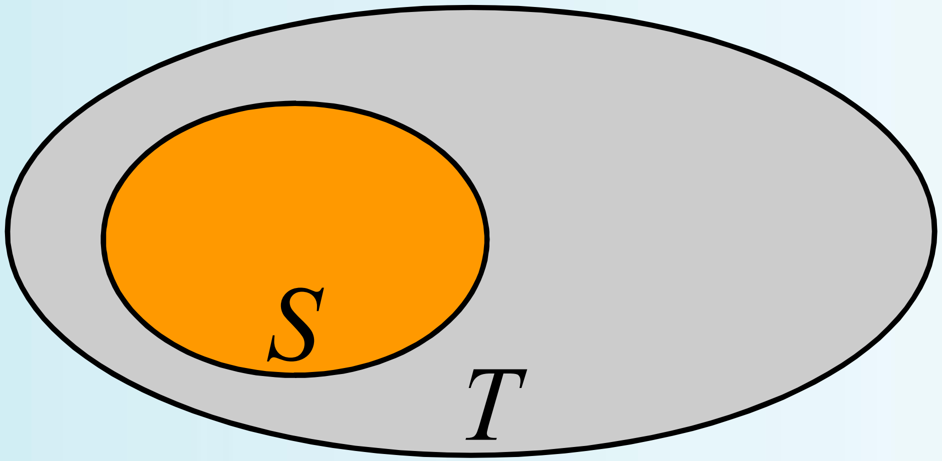
  $\{1, 2, 3, 4\} =$
  $\{x \mid x$ is an integer where $x>0$ and $x < 5\} =$
  $\{x \mid x$ is a positive integer whose square is > 0 and $< 25\}$

Proper Subset and Superset

- *Definition*:

  Let S and T be any two sets. S is a proper subset of T (T is a proper superset of S), denoted by $S \subset T$, *if and only if* $S \subseteq T$ and $S \neq T$.

Example:
$\{1,2\} \subset \{1,2,3\}$



Venn Diagram equivalent of $S \subset T$

Sets are objects, too!

- The objects that are elements of a set may *themselves* be sets.

- Example:

  Let $S=\{x \mid x \subseteq \{1,2,3\}\}$.
  Then $S=\{\emptyset,$
  $\{1\}, \{2\}, \{3\},$
  $\{1,2\}, \{1,3\}, \{2,3\},$
  $\{1,2,3\}\}$

- Note that $1 \neq \{1\} \neq \{\{1\}\}$.

Element of (Member of)

- *Definition*:
  1. $x \in S$ ("x is in S") is the proposition that object x is an *element* or *member* of set S.
     - Example:

       $3 \in \mathbf{N}$, "a" $\in \{x \mid x$ is a letter of the alphabet$\}$

  2. $x \notin S = \neg(x \in S)$ "x is not in S"

Cardinality and Finiteness

- The *cardinality* of S, denoted by $|S|$, is a measure of how many different elements S has.

- Example:
  $|\varnothing|=0$, $|\{1,2,3\}| = 3$, $|\{a,b\}| = 2$, $|\{\{1,2,3\},\{5\}\}| = 2$.

- If $|S| \in \boldsymbol{N}$, then S is said to be *finite*.
  Otherwise, S is said to be *infinite*.

Power Set

- *Definition*:

  Let S be a set. The *power set* $\wp(S)$ of S is the set of all subsets of S, i.e., $\wp(S) = \{x \mid x \subseteq S\}$.

- Example: $\wp(\{a,b\}) = \{\emptyset, \{a\}, \{b\}, \{a,b\}\}$.
- Sometimes $\wp(S)$ is written $\mathbf{2}^S$.
- Note that for finite S, $|\wp(S)| = 2^{|S|}$.
- It turns out that $|\wp(\mathbf{N})| > |\mathbf{N}|$.

There are different sizes of infinite sets where ***N*** *is a set of all natural numbers.*

Ordered n-tuples

- *Definition*:

  For $n \in \boldsymbol{N}$, an *ordered n-tuple* or a *sequence of length n* is defined to be $(a_1, a_2, \ldots, a_n)$. The *first* element is a_1, *etc.*

- These are like sets, except that duplicates matter and the order makes a difference.
- Note $(1, 2) \neq (2, 1) \neq (2, 1, 1)$.
- Empty sequence, singlets, pairs, triples, quadruples, quin<u>tuples</u>, …, n-tuples.

Cartesian Products of Sets

- *Definition*:

  Let A and B be any two sets.

  The *Cartesian product* $A \times B$ is defined to be

  $A \times B = \{(a, b) \mid a \in A \wedge b \in B\}$.

- Example:

  $\{a, b\} \times \{1,2\} = \{(a,1), (a, 2), (b,1), (b, 2)\}$

- Note that for two finite sets, A and B,

  1. $|A \times B| = |A||B|$.
  2. $A \times B \neq B \times A$.

Union Operator

- *Definition*:

  Let A and B be any two sets. The *union* $A \cup B$ of A and B is the set containing all elements that are either in A, or in B (or, of course, in both), i.e.,

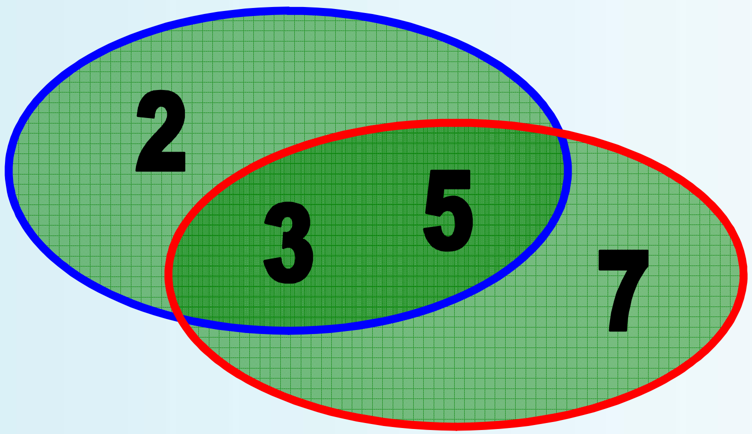
  $A \cup B = \{x \mid x \in A \vee x \in B\}$.

- Note that $A \cup B$ contains all the elements of A and it contains all the elements of B:

  $(A \cup B \supseteq A) \wedge (A \cup B \supseteq B)$

Example of Union

- $\{a,b,c\} \cup \{2,3\} = \{a,b,c,2,3\}$
- $\{2,3,5\} \cup \{3,5,7\} = \{2,3,5,3,5,7\} = \{2,3,5,7\}$



Intersection Operator

- *Definition*:

  Let A and B be any two sets. The *intersection* $A \cap B$ of A and B is the set containing all elements that are simultaneously in A and in B, i.e.,

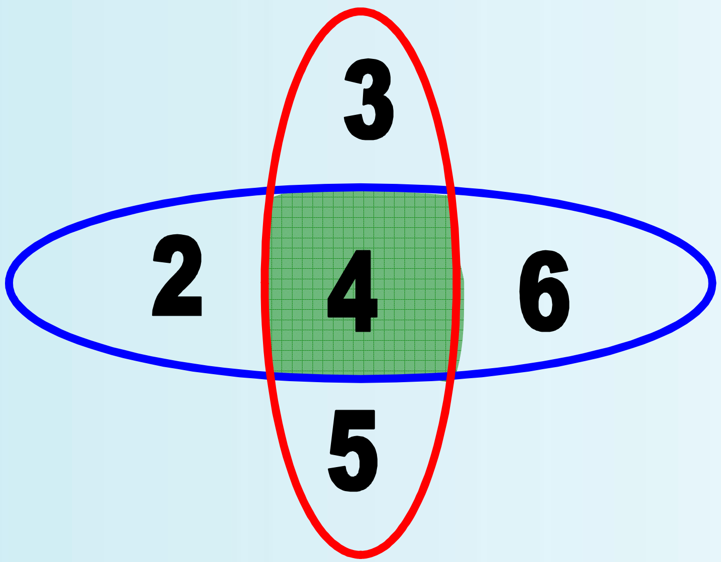
  $A \cap B = \{x \mid x \in A \wedge x \in B\}$.

- Note that $A \cap B$ is a subset of A and it is a subset of B:

  $(A \cap B \subseteq A) \wedge (A \cap B \subseteq B)$

Example of Intersection

- $\{a,b,c\} \cap \{2,3\} = \varnothing$
- $\{2,4,6\} \cap \{3,4,5\} = \{4\}$



Disjointedness

- *Definition*:

  Let A and B be any two sets. A and B are called *disjoint if and only if* their intersection is empty ($A \cap B = \emptyset$).

- Example:

  The set of even integers is disjoint with the set of odd integers.

Inclusion-Exclusion Principle

- How many elements are in $A \cup B$?
  $|A \cup B| = |A| + |B| - |A \cap B|$.

- Example:

  How many students are on our class email list?

  Consider a set $E = I \cup M$ where
  $I = \{s \mid s$ turned in an information sheet$\}$ and
  $M = \{s \mid s$ sent the TAs their email address$\}$.

  Since some students did both,
  $|E| = |I \cup M| = |I| + |M| - |I \cap M|$

Set Difference

- *Definition*:

  Let A and B be any two sets.

  1. The set *difference, $A-B$, of A and B* is the set of all elements that are in A but not in B.
  2. $A-B$ is also called the *complement of B with respect to A*.

Example

1. $\{1,2,3,4,5,6\} - \{2,3,5,7,9,11\} =$ **$\{1,4,6\}$**
2. $\boldsymbol{Z} - \boldsymbol{N} = \{\ldots, -1, 0, 1, 2, \ldots\} - \{1, \ldots\}$
   $= \{x \mid x \text{ is an integer but not a nat. number}\}$
   $= \{x \mid x \text{ is a negative integer or } x=0\}$
   $= \{\ldots, -3, -2, -1, 0\}$

Universal Set & Complement of a Set

- *Definition* (Universal Set):

  A set is a universal set or a universe of discourse, denoted by U, if it includes every set under discussion.

- *Definition* (Complement of a Set):

  Let A be a set. The *complement* of A in U, denoted by $\overline{A}$, is the set of all elements of U which are not elements of A, i.e.,

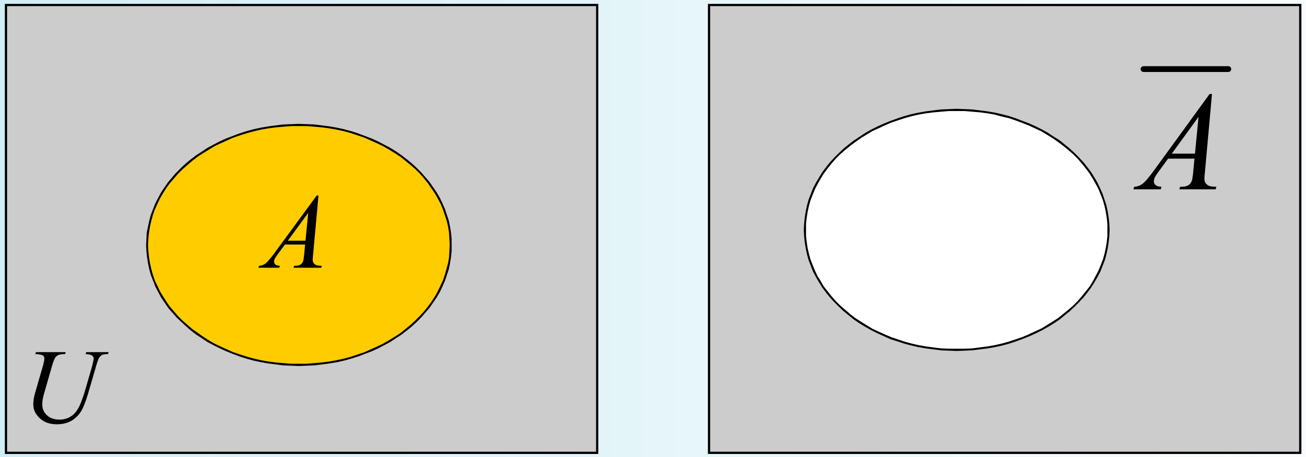
  $\overline{A} = U - A.$

  Example:

  If U=$\mathbf{N}$, $\overline{\{3,5\}} = \{1,2,4,6,7,...\}$

- An equivalent definition, when *U* is clear:

$$\overline{A} = \{x \mid x \notin A\}$$



Set Identity Theorems

For any sets, *A*, *B*, and *C*, the following holds:

1. *Identity*: $A \cup \varnothing = A, \quad A \cap U = A$
2. *Domination*: $A \cup U = U, \quad A \cap \varnothing = \varnothing$
3. *Idempotent*: $A \cup A = A = A \cap A$
4. *Double complement*: $\overline{(\overline{A})} = A$
5. Commutative: $A \cup B = B \cup A, \quad A \cap B = B \cap A$
6. *Associative*: $A \cup (B \cup C) = (A \cup B) \cup C$
   $A \cap (B \cap C) = (A \cap B) \cap C$

DeMorgan's Theorem for Sets

- *Theorem*:

  Let A and B be sets. Then the following holds:

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

Example:

Let A, B, and C be sets. Show that $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Proof:

1. Show $A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$:

   Let $x \in A \cap (B \cup C)$. Then by definition of $\cap$, $x \in A$ and $x \in (B \cup C)$.

   By definition of $\cup$, $x \in B$ or $x \in C$.

   Case 1: Let $x \in B$. Then by definition of $\cap$, $x \in A \cap B$.

   By definition of by $\cup$, $x \in (A \cap B) \cup (A \cap C)$.

   Case 2: Let $x \in C$. Then by definition of $\cap$, $x \in A \cap C$.

   By definition of by $\cup$, $x \in (A \cap B) \cup (A \cap C)$.

   From case 1 and 2, $x \in (A \cap B) \cup (A \cap C)$.

   By definition of $\subseteq$, $A \cap (B \cup C) \subseteq (A \cap B) \cup (A \cap C)$.

2. Show $(A \cap B) \cup (A \cap C) \subseteq A \cap (B \cup C)$: Similarly done.

From 1 and 2, $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ by definition of set equality.

- Theorem:

  If A and B are two sets, the following statements are equivalent.

  (1) $A \subseteq B$

  (2) $A \cap B = A$

  (3) $A \cup B = B$

Generalized Unions & Intersections

- Since union & intersection are commutative and associative, we can extend them from operating on *ordered pairs* of sets (A, B) to operating on sequences of sets $(A_1, \ldots, A_n)$, or even unordered *sets* of sets.

Generalized Union

1. Binary union operator: $A \cup B$
2. n-ary union:
   $A_1 \cup A_2 \cup \ldots \cup A_n = ((\ldots((A_1 \cup A_2) \cup \ldots) \cup A_n)$
   (grouping & order is irrelevant)
3. "Big $\cup$" notation: $\bigcup_{i=1}^{n} A_i$
4. For infinite sets of sets: $\bigcup_{A \in X} A$

Generalized Intersection

1. Binary intersection operator: $A \cap B$
2. n-ary intersection:
   $A_1 \cap A_2 \cap \ldots \cap A_n \equiv ((\ldots((A_1 \cap A_2) \cap \ldots) \cap A_n)$
   (grouping & order is irrelevant)
3. "Big ∩" notation: $\bigcap_{i=1}^{n} A_i$
4. For infinite sets of sets: $\bigcap_{A \in X} A$

Exercise

1. Let A and B be sets. Show that

   (a) $(A \cap B) \subseteq A$

   (b) $A \cup (B\text{-}A) = A \cup B$

   (c) $A \cap B = A$ if and only if $A \cup B = B$

   (d) $A\text{-} (A\cap B) = A\text{-}B$

   (e) $\neg(A \cup B) = \neg A \cap \neg B$

2. Let A, B and C be sets. Show that

   $(A\text{-}B)\text{-}C = (A\text{-}C)\text{-}(B\text{-}C)$.

3. Let A and B be two sets. Prove or disprove each of the followings:

(a) $\wp(A) \cup \wp(B) \subseteq \wp(A \cup B)$ where $\wp(A)$ is the power set of the set A.

(b) $\wp(A \cup B) \subseteq \wp(A) \cup \wp(B)$

4. Which of the following are true for all sets, A, B, and C ? Give a counter example if the answer is false (No proof is necessary if the answer is true).

(a) If $A \cap B = \emptyset$ and $B \cap C = \emptyset$, then $A \cap C = \emptyset$.

(b) If $A \in B$ and $\neg(B \subseteq C)$, then $\neg(A \in C)$.

(c) If $A \in B$ and $B \in C$, then $\neg(A \in C)$.

(d) $(A \cap B) \cup C = A \cap (B \cup C)$ if and only if $C \subseteq A$.

(e) $\emptyset \in A$.

(f) If $A \subseteq B$ and $B \in C$, then $A \subseteq C$

(g) If $A \in B$, then $\{A\} \subseteq B$

Discrete Mathematics

3. Relations

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

Binary Relations

- *Definition*:

  Let A and B be any two sets. A *binary relation* R from A to B is a subset of $A \times B$.

- The notation aRb means $(a,b) \in R$.
  - Example:

    $$a \leq b \text{ means } (a,b) \in \leq$$

    where $\leq$ denotes the relation of *partial ordering*.

Complementary Relations

- *Definition*:

  Let $R \subseteq A \times B$ be any binary relation. Then, $\overline{R}$, the *complement* of R, is the binary relation defined by

  $$\overline{R} = \{(a,b) \mid (a,b) \notin R\} = (A \times B) - R$$

- Note that the complement of $\overline{R}$ is R.

Inverse Relations

- *Definition*:

  An inverse relation of a binary relation $R \subseteq A \times B$, denoted by R^{-1}, is defined to be

  $R^{-1} = \{(b, a) \mid (a, b) \in R\}$.

- *Theorem*:

1. $(R_1 \cup R_2)^{-1} = R_1^{-1} \cup R_2^{-1}$
2. $(R_1 \cap R_2)^{-1} = R_1^{-1} \cap R_2^{-1}$

Relations on a Set

- *Definition*:
  1. A (binary) relation from a set A to itself is called a relation *on* the set A.
  2. The *identity relation* $\boldsymbol{I}_A$ on a set A is the set,
     $\boldsymbol{I}_A = \{(a,a) \mid a \in A\}$.

Properties of Relations

- *Definition*:

1. A relation R on A is *reflexive* if for every a in A, $(a, a) \in R$.
2. A relation R on A is *irreflexive* if for every a in A, $(a, a) \notin R$.
3. A relation R on A is *symmetric* if for every a and b in A, if $(a,b) \in R$, then $(b,a) \in R$.
4. A relation R on A is *antisymmetric* if for every a and b in A, if $(a,b) \in R$ and $(b,a) \in R$, then $(a=b)$.
5. A relation R on A is *asymmetric* if for every a and b in A, if $(a,b) \in R$, then $(b,a) \notin R$.
6. A relation R on A is *transitive* if for every a, b, and c in A, if $(a,b) \in R$ and $(b,c) \in R$, then $(a,c) \in R$.

- Note "*irreflexive*" ≠ "*not reflexive*"!

Composite Relations

- *Definition*:

  Let $R \subseteq A \times B$, and $S \subseteq B \times C$. Then the *composite* of R and S, denoted by $R \circ S$, is defined to be

  $R \circ S = \{(a,c) \mid (a,b) \in R \wedge (b,c) \in S$ for some b in $B\}$

- *Definition*:

  The n^{th} power R^n of a relation R on a set A can be defined recursively by $R^{n+1} = R^n \circ R$ for all $n \geq 0$ where $R^0 = \boldsymbol{I}_A$.

- *Theorem*:

  Let R_1, R_2, and R_3 be relations on a set A. Then

  1. $R_1 \circ (R_2 \cap R_3) \subseteq (R_1 \circ R_2) \cap (R_1 \circ R_3)$
  2. $R_1 \circ (R_2 \cup R_3) = (R_1 \circ R_2) \cup (R_1 \circ R_3)$

- *Theorem*:

  Let R be a relation on a set A, i.e. $R \subseteq A \times A$, and I_A be a identity relation on a set A, ($I_A = \{<x,x> | x \in A\}$).
  Then the following holds:

  1. R is *reflexive* iff $I_A \subseteq R$
  2. R is *irreflexive* iff $I_A \cap R = \varnothing$
  3. R is *symmetric* iff $R = R^{-1}$
  4. R is *asymmetric* iff $R \cap R^{-1} = \varnothing$
  5. R is *antisymmetric* iff $R \cap R^{-1} \subseteq I_A$
  6. R is *transitive* iff $R \circ R \subseteq R$

Walk, path, cycle, loop, sling

- *Definition*:

  Given a directed graph G=<N, V> where N is a set of nodes and V is a set of edges,

  1. A *walk* is a sequence $x_0, x_1, \ldots, x_n$ of the vertices of a directed graph such that $x_i x_{i+1}$, $0 \le i \le n$-1, is an edge.
  2. The *length of a walk* is the number of edges in the walk.
  3. If a walk holds $x_i \neq x_j$ $(i \neq j)$ i, j =0, …, n, (i.e., no edge is repeated), the walk is called a *path*.
  4. If a walk holds $x_i \neq x_j$ $(i \neq j)$ i, j =0, …, n, except $x_0 = x_n$, the walk is called a *cycle*.
  5. A *loop* is a cycle of length one.
  6. A *sling* is a cycle of length two.

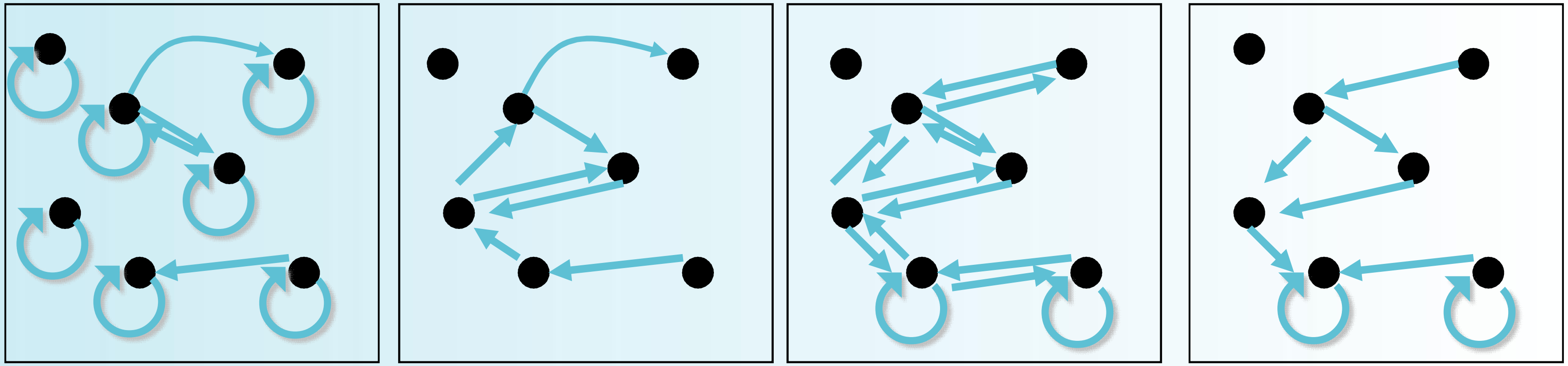
- *Theorem*:

  Given a directed graph G=<N, R> where N is a set of nodes and R is a set of edges,

  1. R is *reflexive* iff G has a loop at every node.
  2. R is *irreflexive* iff G has no loop at any node.
  3. R is *symmetric* iff if G has a walk of length one between two distinct nodes, then it has a sling between them.
  4. R is *asymmetric* iff if G has a walk of length one between two distinct nodes, then it has no sling between them and no loop at any node.
  5. R is *antisymmetric* iff if G has a walk of length one between two distinct nodes, then it has no sling between them.
  6. R is *transitive* iff if G has a walk of length two between two nodes, then it has a walk of length one between them.

Digraph Reflexive, Symmetric

It is extremely easy to recognize the reflexive/irreflexive/ symmetric/antisymmetric properties by graph inspection.



Reflexive: Every node has a loop

Irreflexive: No node has a loop

Symmetric: Every link is bidirectional

Antisymmetric: No link is bidirectional

Closures of Relations

- *Definition*:

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

- *Theorem*:
  1. The *reflexive closure* of a relation R on A is obtained by adding (a,a) to R for each $a \in A$, *i.e.*, $r(R) = R \cup I_A$.
  2. The *symmetric closure* of R is obtained by adding (b,a) to R for each (a,b) in R, *i.e.*, $s(R) = R \cup R^{-1}$.
  3. The *transitive closure* or *connectivity relation* of R is obtained by repeatedly adding (a,c) to R for each $(a,b),(b,c)$ in R, *i.e.*,

     $$t(R) = \bigcup_{n \in \mathbf{Z}^+} R^n$$

Equivalence Relations

- *Definition*:

  A relation R on a set A is called an *equivalence relation* if it is *reflexive, symmetric,* and *transitive.*

Equivalence Classes

- *Definition*:

  Let R be any equivalence relation on a set A. For each a in A, the *equivalence class* of a with respect to R, denoted by $[a]_{R,}$ is

$$[a]_R = \{ b \mid \langle a,b\rangle \in R\}$$

- Examples:
  1. "Strings a and b are the same length."
     - $[a]$ = the set of all strings of the same length as a.
  2. "Integers a and b have the same absolute value."
     - $[a]$ = the set $\{a, -a\}$
  3. "Real numbers a and b have the same fractional part (*i.e.*, $a - b \in \boldsymbol{Z}$)."
     - $[a]$ = the set $\{\ldots, a-2, a-1, a, a+1, a+2, \ldots\}$
  4. "Integers a and b have the same residue modulo m." (for a given $m>1$)
     - $[a]$ = the set $\{\ldots, a-2m, a-m, a, a+m, a+2m, \ldots\}$

- *Theorem*:

  Let R be an equivalence relation on a set A.

  1. For every x in A, $x \in [x]_R$.
  2. If $<x, y> \in R$, then $[x]_R=[y]_R$.

- *Theorem*:

  Let R be an equivalence relation on a set A.

  If $<x, y> \notin R$, then $[x]_R \cap [y]_R = \varnothing$.

Partition and Covering of a Set

- *Definition*:

  Let S be a give set and $A = \{A_1, A_2, \ldots, A_m\}$ where each A_i, i=1, … m, is a non-empty subset of S and

  $$\bigcup_{i=1}^{m} A_i = S.$$

  1. Then the set A is called a *covering* of S, and the sets $A_1, A_2, \ldots, A_m$ are said to *cover S*.
  2. If the elements of A, which are subsets of S, are mutually disjoint, then A is called a *partition* of S, and the sets $A_1, A_2, \ldots, A_m$ are called the *blocks* of the partition.

Refinement and a Quotient Set

- *Definition*:

  Let R be an equivalence relation on a set A, then $A/R = \{[x]_R | x \in A\}$ is called a *quotient set of A modulo R.*

- *Theorem*:

  Let R be an equivalence relation on a set A, then the quotient set of A modulo R is a partition of A.

Relation induced by the Partition

- *Definition*:

  Let A be a set. Let $\pi = \{A_1, A_2, \ldots, A_n\}$ be a partition of A. R_π is a *relation induced by the partition* π and defined as follows.

  $R_\pi = \{\langle x, y \rangle \mid (x \in A_i) \wedge (y \in A_i) \text{ for some } i\}$

- *Theorem*:

  Let A be a set. Let $\pi = \{A_1, A_2, \ldots, A_n\}$ be a partition A and R_π be the relation induced by the partition π. Then, R_π is an equivalence relation on A.

Refinement

- *Definition*:

  Let π_1 and π_2 be two partitions of a set A. π_2 is a *refinement* of π_1, (π_2 refines π_1), if for every block B_i in π_2, there exists some block A_j in π_1 such that $B_i \subseteq A_j$.

- *Theorem*:

  Let π and π' be two partitions of a nonempty set A and let R_π and $R_{\pi'}$ be the equivalence relations induced by π and π' respectively. Then π' refines π if and only if $R_{\pi'} \subseteq R_\pi$.

Partial Orderings

- *Definition*:

1. A relation R on a set S is called a *partial ordering* or *partial order* iff it is *reflexive, antisymmetric*, and *transitive*.
2. A set S together with a *partial ordering* R is called a *partially ordered set*, or *poset*, denoted by (S, R).

- Example:

  Consider the "greater than or equal to" relation $\geq$ (defined by $\{(a, b) \mid a \geq b\}$). Is $\geq$ a partial ordering on the set of integers?

  Proof:

  1. $\geq$ is reflexive, because $a \geq a$ for every integer a.
  2. $\geq$ is antisymmetric, because if $a \geq b \wedge b \geq a$, then $a=b$.
  3. $\geq$ is transitive, because if $a \geq b$ and $b \geq c$, then $a \geq c$.

  Consequently, $(\boldsymbol{Z}, \geq)$ is a partially ordered set.

- Example:

  Is the “inclusion relation” $\subseteq$ on the power set of a set S a partial ordering ?

  Proof:

  1. $\subseteq$ is reflexive, because $A \subseteq A$ for every set A.
  2. $\subseteq$ is antisymmetric, because if $A \subseteq B \wedge B \subseteq A$, then $A = B$.
  3. $\subseteq$ is transitive, because if $A \subseteq B$ and $B \subseteq C$, then $A \subseteq C$.

  Consequently, $(\wp(S), \subseteq)$ is a partially ordered set

Partially Ordered Sets

- In a poset the notation $a \leq b$ denotes that $(a, b) \in \leq$.

Note that the symbol $\leq$ is used to denote the relation in any poset, not just the "less than or equal" relation. The notation $a < b$ denotes that $a \leq b$, but $a \neq b$. If $a < b$ we say "a is less than b" or "b is greater than a".

• For two elements a and b of a poset $(S, \leq)$, it is possible that neither $a \leq b$ nor $b \leq a$. For instance, in $(\wp(\boldsymbol{Z}), \subseteq)$, $\{1, 2\}$ is not related to $\{1, 3\}$, and vice versa, since neither is contained within the other.

- *Definition*:

    1. The elements a and b of a poset $(S, \leq)$ are called comparable if either $a \leq b$ or $b \leq a$.

    2. The elements a and b of a poset $(S, \leq)$ are called incomparable if neither $a \leq b$ nor $b \leq a$.

- *Definition* :

  If $(S, \leq)$ is a poset and every two elements of S are comparable, $(S, \leq)$ is called a *totally ordered* or *linearly ordered set*, and $\leq$ is called a *total order* or *linear order*. A totally ordered set is also called a *chain*.

- Example 1**:** Is (**Z**, $\leq$) a totally ordered poset?

Yes, because a $\leq$ b or b $\leq$ a for all integers a and b.

- Example 2**:** Is ($\mathbf{Z}^+$, |) a totally ordered poset?

No, because it contains incomparable elements such as 5 and 7.

Hasse Diagram

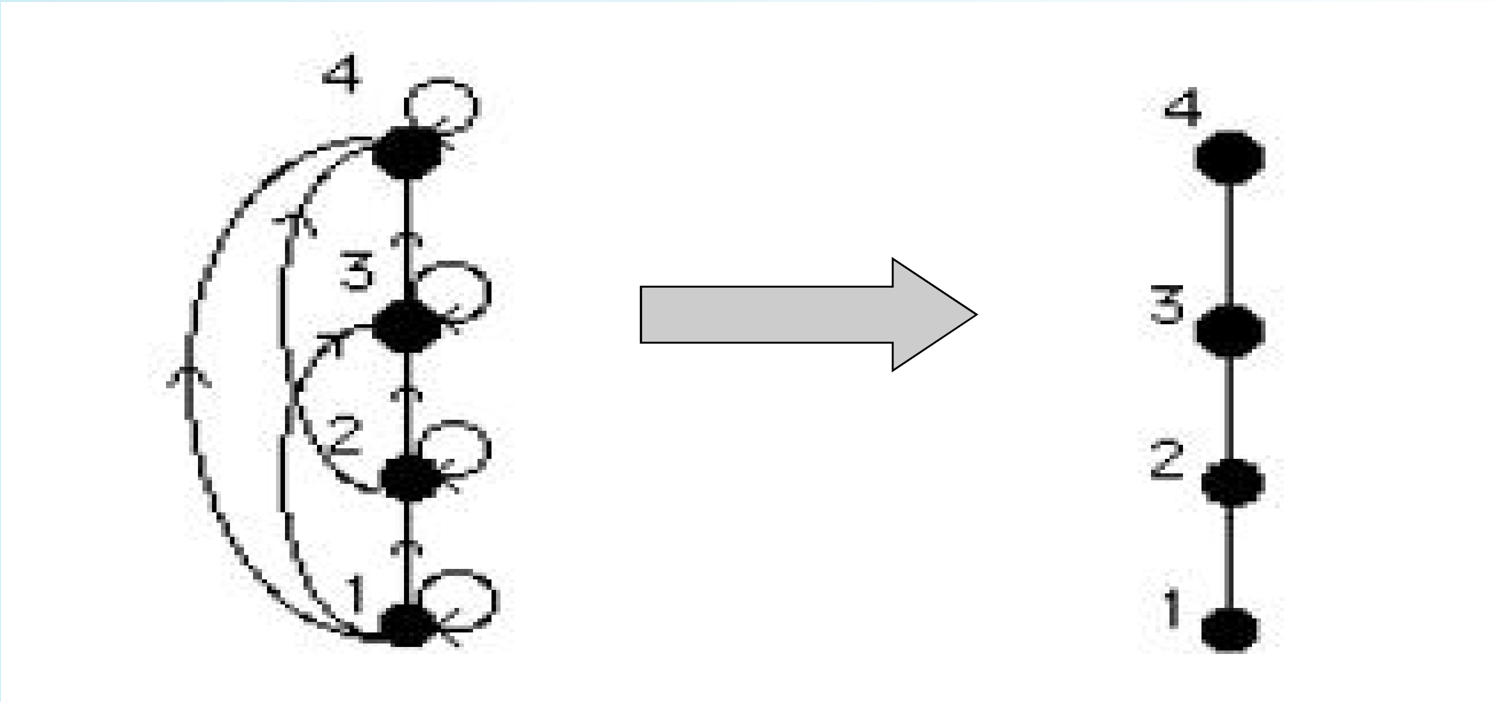
- *Definition* **:**

  Let G be a digraph representing a poset, $(A, \leq)$.
  The Hasse diagram of $(A, \leq)$ is constructed from G by

  1. All loops are omitted.
  2. An arc is not present in a Hasse diagram if it is implied by the transitivity of the relation.
  3. All arcs point upward and arrow heads are not used.

Example of Hasse Diagram

- $\{ <a, b> \mid a \le b \}$ on $\{1, 2, 3, 4\}$



Greatest Elements and Least Elements

- *Definition*:

  Let $(A, \leq)$ be a poset and B be a subset of A.

  1. An element $a \in B$ is *a greatest element* of B iff for every element $a' \in B$, $a' \leq a$.
  2. An element $a \in B$ is *a least element* of B iff for every element $a' \in B$, $a \leq a'$.

- *Theorem*:

  Let $(A, \leq)$ be a poset and $B \subseteq A$. if a and b are greatest (least) elements of B, then a=b

Least Upper Bound (lub)

- *Definition*:

  Let $(A, \leq)$ be a poset and B be a subset of A.

  1. An element $a \in A$ is an *upper bound* for B iff for every element $a' \in B$, $a' \leq a$.
  2. An element $a \in A$ is a *least upper bound (lub)* for B iff a is an upper bound for B and for every upper bound a' for B, $a \leq a'$.

Greatest Lower Bound (glb)

- *Definition*:

  Let $(A, \leq)$ be a poset and B be a subset of A.

  1. An element $a \in A$ is a *lower bound* for B iff for every element $a' \in B$, $a \leq a'$.
  2. An element $a \in A$ is a *greatest lower bound (glb)* for B iff a is a lower bound for B and for every lower bound a' for B, $a' \leq a$.

lub and glb

- *Theorem*:

  Let $(A, \leq)$ be a poset and $B \subseteq A$.

  1. If b is a greatest element of B, then b is a lub of B.
  2. If b is an upper bound of B and $b \in B$, then b is a greatest element of B.

- *Theorem*:

  Let $(A, \leq)$ be a poset and $B \subseteq A$.

  If a least upper bound (or a greatest lower bound) for B exists, then it is unique.

Lattices

- *Definition*:

  A poset is a *lattice* if every pair of elements has a *lub* and a *glb*.

- *Theorem*:

  Let $<L, \leq>$ be a lattice. If $x*y$ ($x+y$) denotes the *glb* (*lub*) for $\{x, y\}$, then the following holds: for any a, b, and c in L,

(i) $a*a=a$	(i') $a+a=a$	(*idempotent*)
(ii) $a*b=b*a$	(ii') $a+b=b+a$	(*commutative*)
(iii) $(a*b)*c= a*(b*c)$	(iii') $(a+b)+c= a+(b+c)$	(*associative*)
(iv) $a*(a+b)=a$	(iv') $a+(a*b)=a$	(*absorption*)

Exercise

1. For each of the following relation R on set A, state whether or not R is *reflexive, irreflexive, symmetric, asymmetric, antisymmetric, and transitive.*
   (a) $A = \{1, 2, \ldots., 9\}$
   $R = \{<x,y> \mid x+y=10\}$
   (b) A = a set of real numbers
   $R = \{<x,y> \mid |x| \leq |y| \}$
   (c) A = a set of natural numbers
   $\mathrm{R} = \{<x,y> \mid x\text{-}y=2k,\ k \in A \}$

2. Suppose that R and S are reflexive relations on a set A.
   Prove or disprove each of theses statements
   (a) $R \cup S$ is reflexive
   (b) $R \cap S$ is reflexive

3. Show that the relation R on a set A is symmetric if and only if $R=R^{-1}$, where R^{-1} is the inverse relation.

4. Let R_1 and R_2 be arbitrary relations on a set A.
Prove or disprove the following assertions.
(a) If R_1 and R_2 are reflexive, then $R_1 \circ R_2$ is reflexive.
(b) If R_1 and R_2 are transitive, then $R_1 \circ R_2$ is transitive.
(c) If R_1 and R_2 are symmetric, then $R_1 \circ R_2$ is symmetric.

5. Show that the relation R on a set A is symmetric if and only if $R=R^{-1}$, where R^{-1} is the inverse relation.

6. Let A be a set of ordered pairs of positive integers and R be a relation on A such that $<(x,y),(u,v)> \in R$ if and only if $x+v = y+u$. Determine whether or not R is an equivalence relation.

7. Let R_1 and R_2 be two equivalence relations on a nonempty set A. Prove or disprove the following :

   (a) $R_1 \cup R_2$ an equivalence relation.

   (b) $R_1 \cap R_2$ an equivalence relation.

8. If R is a partial ordering relation on a set X and $A \subseteq X$, show that $R \cap (A \times A)$ is a partial ordering on A.

9. Let S be a set of all partitions defined on a nonempty set A. The relation R on a set S is defined to be $<\pi_1, \pi_2> \in R$ if and only if π_1 refines π_2 (π_1 is the refinement of π_2).

(a) Show that R is a partial ordering.

(b) Is a p.o. set $<S, R>$ a lattice? If yes, prove it. Otherwise, explain why.

10. Let $<A, \leq>$ be a lattice. Prove that for every x, y, and z in A,

(a) $x*(y*z) = (x*y)*z$

(b) $x+(x*y) = x$

where $x*y$ is glb(x,y) and $x+y$ is lub(x,y).

11. Let $<E(A), \subseteq>$ be a p.o.set where $E(A)$ is a set of all equivalence relations defined on a set A.

(a) For every x and y in $E(A)$, is $x \cap y$ the glb of $\{x,y\}$?

(b) For every x and y in $E(A)$, is $x \cup y$ the lub of $\{x,y\}$?

Discrete Mathematics

4. Functions

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

Functions

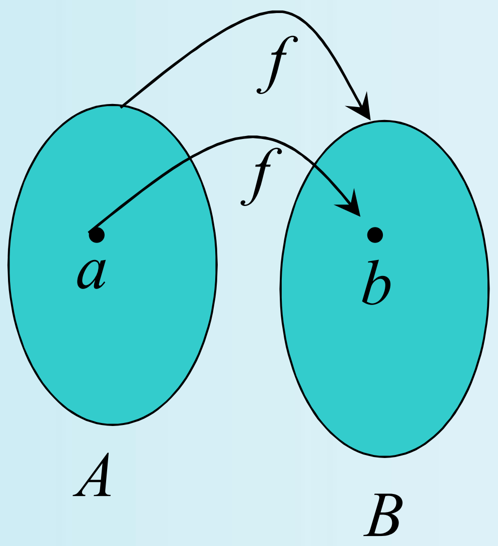
- *Definition*:

  Let A and B be two sets. A relation f from A to B is called a function if for every x in A, there is a unique y in B such that $<x, y> \in f$

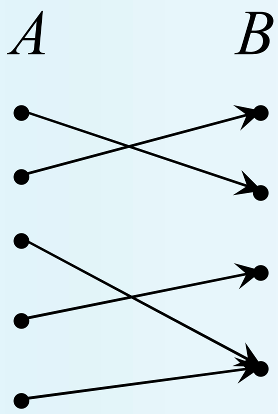
- A function $f \subseteq A \times B$ may be written by $f : A \rightarrow B$ and $<x, y> \in f$ written by $f(x)=y$.

Graphical Representations

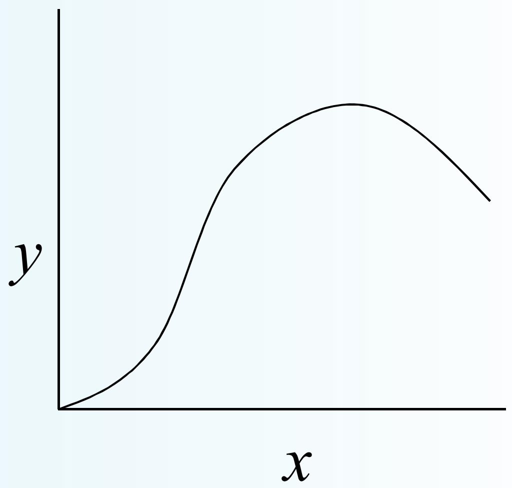
- Functions can be represented graphically in several ways:



Like Venn diagrams



Bipartite Graph



Plot

Some Function Terminology

- *Definition*:

  Let f:A→B and $f(a)=b$ (where a in A and b in B). Then,

  1. A is the *domain* of f.
  2. B is the *codomain* of f.
  3. b is the *image* of a under f.
  4. a is a *pre-image* of b under f.
  5. The *range* $R \subseteq B$ of f is $R = \{b \mid (a, b) \in f$ for some $a\}$.

Images of Sets under Functions

- *Definition*:

  Given $f{:}A \rightarrow B$, and $S \subseteq A$, the *image* of S under f is defined to be the set of all images (under f) of the elements of S: $f(S) = \{\, f(w) \mid w \in S\}$

- Note the range of f can be defined as simply the image (under f) of f's domain!

Range versus Codomain

- The range of a function might *not* be its whole codomain.
- The codomain is the set that the function is *declared* to map all domain values into.
- The range is the *particular* set of values in the codomain that the function *actually* maps elements of the domain to.

Range vs. Codomain - Example

- Suppose I declare to you that: "f is a function mapping students in this class to the set of grades $\{A,B,C,D,E\}$."
- At this point, you know f's codomain is: $\{A,B,C,D,E\}$, and its range is unknown!
- Suppose the grades turn out all As and Bs.
- Then the range of f is $\{A,B\}$, but its codomain is still $\{A,B,C,D,E\}$!

Restriction and Extension

- *Definition*:

  If $f: X \to Y$ and $A \subseteq X$, then $f \cap (A \times Y)$ is a function from A to Y called the *restriction* of f to A and is sometimes written as f/A, If g is a restriction of f, then f is called the *extension* of g.

Operators

- *Definition*:

  An n-ary *operator* O_n over the set S is a function from the set of ordered n-tuples of elements of S to S itself.

$$O_n : S^n \rightarrow S$$

- Example:

1. If $S=\{\mathbf{T},\mathbf{F}\}$, $\neg$ can be seen as a unary operator, and $\wedge,\vee$ are binary operators on S.
2. $\cup$ and $\cap$ are binary operators on the set of all sets.

Function Operators

- If $\bullet$ ("dot") is any operator over B, then we can extend $\bullet$ to also denote an operator over functions $f{:}A\rightarrow B$.

- *Definition*:

  Given any binary operator $\bullet{:}B\times B\rightarrow B$ and two functions, $f{:}A\rightarrow B$ and $g{:}A\rightarrow B$,

  the function, $(f \bullet g){:}A\rightarrow B$, is defined to be such that $\forall a\in A, (f \bullet g)(a) = f(a)\bullet g(a)$.

Example

- Let + and $\times$ be addition and multiplication (binary) operators over $\boldsymbol{R}$, respectively. Then, two functions, f:$\boldsymbol{R}\rightarrow\boldsymbol{R}$ and g:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, can be also *added* and *multiplied*:
  1. $(f + g)$:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, where $(f + g)(x) = f(x) + g(x)$
  2. $(f \times g)$:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, where $(f \times g)(x) = f(x) \times g(x)$

Function Composition

- *Definition*:

  Let $g{:}A \rightarrow B$ and $f{:}B \rightarrow C$ be two functions. Then the function composition, $f \circ g$, from A to C is

  $f \circ g = \{ <x,y> \mid (\exists z)((<x,z> \in g) \wedge (<z,y> \in f))\}$

- Note that $\circ$ (like Cartesian $\times$, but unlike $+, \wedge, \cup$) is not commutative. (Generally, $f \circ g \neq g \circ f$.)

- *Theorem*:

  Let $g:A\rightarrow B$ and $f:B\rightarrow C$ be functions. Then the function composition $f\circ g$ is a function from A to C and $(f\circ g)(a) = f(g(a))$ for all a in A

- *Theorem*:

  Composition of functions is associative: If f, g, and h are functions, then $(f\circ g)\circ h= f\circ(g\circ h)$

Partial Function

- *Definition*:

  Let X and Y be sets. A *partial function* f with domain X and codomain Y is any function from X' to Y, where $X' \subseteq X$.

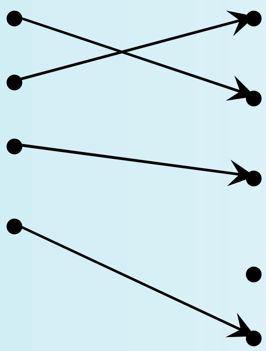
One-to-One (Injective) Functions

- *Definition*:

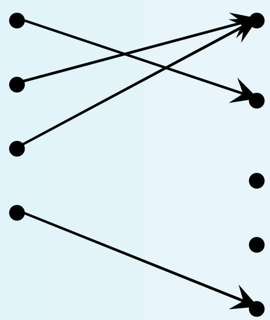
  A function $f : A \rightarrow B$ is *one-to-one*, or *injective*, or *an injection*, if every element of its range has *only* 1 pre-image : (for every x and y in A, if $f(x)=f(y)$, then $x=y$)

Illustration of One-to-One

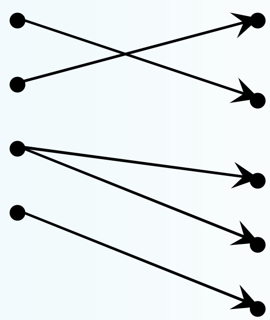
- Bipartite (2-part) graph representations of functions that are (or not) one-to-one:



One-to-one



Not one-to-one



Not even a function!

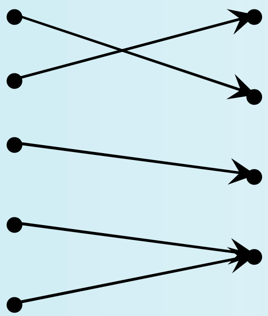
Onto (Surjective) Functions

- *Definition*:

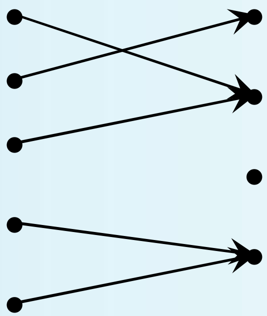
  A function $f : A \rightarrow B$ is *onto* or *surjective* or *a surjection* if for every b in B, there exists a in A such that $f(a)=b$.

Illustration of Onto

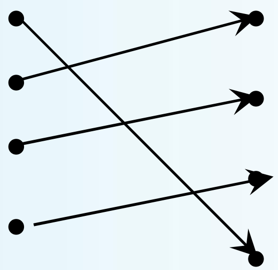
- Some functions that are (or not) *onto* their codomains:



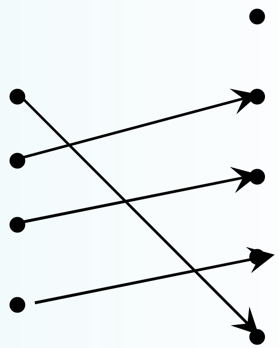
Onto
(but not 1-1)



Not Onto
(not 1-1)



Both 1-1
and onto



1-1 but
not onto

Bijective Functions

- *Definition*:

  A function $f : A \to B$ is *one-to-one and onto,* or *a one-to-one correspondence*, or *bijective*, or *a bijection* if it is both one-to-one and onto.

- *Theorem*:

  Let $f \circ g: A \to C$ be a composite function where $g: A \to B$ and $f: B \to C$.

1. If f and g are *surjective*, then $f \circ g$ is *surjective.*
2. If f and g are *injective*, then $f \circ g$ is *injective*.
3. If f and g are *bijective*, then $f \circ g$ is *bijective*.
4. If $f \circ g$ is *surjective*, then f is *surjective.*
5. If $f \circ g$ is *injective*, then g is *injective*.
6. If $f \circ g$ is bijective, then f is *surjective* and g is *injective.*

Constant Function

- *Definition*:

  Let a function f: $X \rightarrow Y$ is a *constant function* if there exist some y in Y such that $f(x)=y$ for every x in X

Identity Function

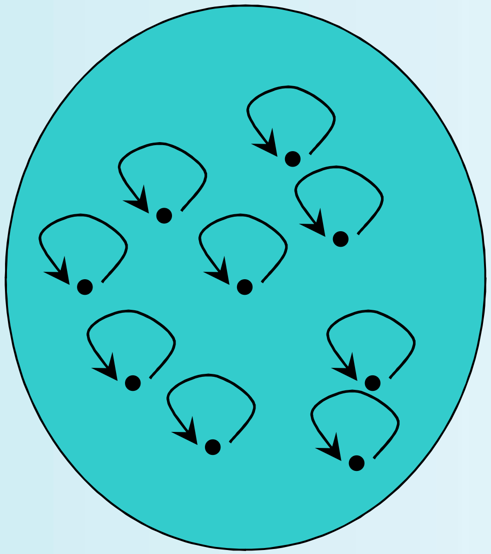
- *Definition*:

  For any domain A, the *identity function* I: $A \rightarrow A$ (variously written, I_A, $\mathbf{1}$, $\mathbf{1}_A$) is the unique function such that for every a in A, $I(a)=a$.

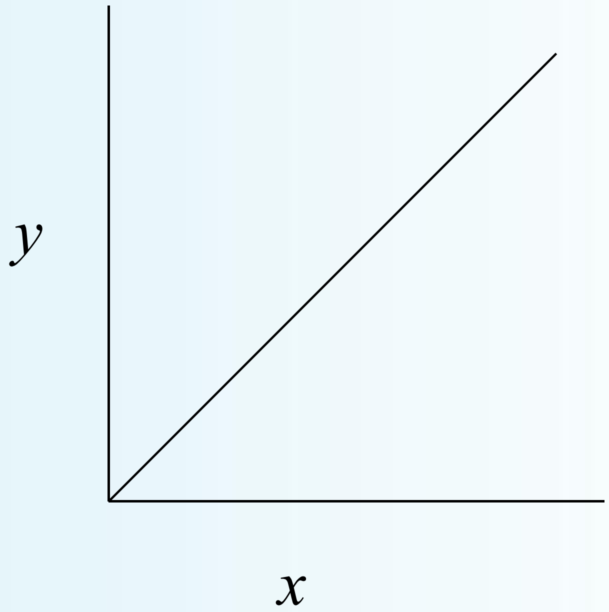
- Note that the identity function is one-to-one and onto (bijective).
- Note that if f: $X \rightarrow Y$, then $f = f \circ I = I \circ f$

Identity Function Illustration

- The identity function:



Domain and range



Inverse Function

- *Definition*:

  Let $f{:}X\rightarrow Y$ be a bijection from X to Y. The *inverse function* of f, denoted by f^{-1}, is the converse relation of f.

- *Theorem*:

  1. Let f be a bijective function $f{:}X\rightarrow Y$. Then f^{-1} is a bijective function, $f^{-1}: Y \rightarrow X$.
  2. If f is bijective, then $(f^{-1})^{-1} = f$.

- *Definition*:

  Let $h:A\rightarrow B$ and $g:B\rightarrow A$. If $g \circ h = I_A$, then g is a *left inverse* of h and h is *a right inverse* of g.

- *Theorem*:

  Let $f:A\rightarrow B$ with $A\neq \emptyset$. Then

  1. f has *a left inverse* if and only if f is *injective.*
  2. f has *a right inverse* if and only if f is *surjective.*
  3. f has *a left* and *a right inverse* if and only if f is *bijective.*
  4. If f is *bijective*, then *the left* and *the right inverse* of f are equal.

A Couple of Key Functions

- In discrete math, we will frequently use the following functions over real numbers:
  1. $\lfloor x \rfloor$ ("floor of x") is the largest (most positive) integer $\leq x$.
  2. $\lceil x \rceil$ ("ceiling of x") is the smallest (most negative) integer $\geq x$.

Finite Set and Cardinality

- *Definition*:

  A set A is *finite* if there is some natural number $n \in N$ such that there is a *bijection* from the set $\{1, 2, \ldots, n\}$ of the first n natural numbers to the set A.

  The integer n is called the *cardinality* of A, and we say "A has n elements," or " n is the *cardinal number* of A." The cardinality of A is denoted by $|A|$. A set is *infinite* if it is not finite.

- *Theorem*:

  Let A and B be finite sets, and suppose there is a bijection from A to B. Then $|A|=|B|$

Countability

- *Definition*:

  A set A is of cardinality $\aleph_0$ denoted $|A| = \aleph_0$ if there is a *bijection* from $\boldsymbol{N}$ to A where $\boldsymbol{N}$ is a set of all natural numbers.

- *Definition*:

  A set A is *countably infinite* if $|A| = \aleph_0$. The set A is *countable* or *denumerable* if it is either finite or countably infinite. The set A is *uncountable* or *uncountably infinite* if it is not countable.

Cardinality

- *Definition*:

  For any two (possibly infinite) sets A and B, we say that A and B *have the same cardinality* (written $|A|=|B|$) if there exists a bijection from A to B.

Countable versus Uncountable

- *Countable*: All elements of S can *be enumerated* in such a way that *any* individual element of S will eventually be *counted* in the enumeration. Examples: $\boldsymbol{N}$, $\boldsymbol{Z}$.

- *Uncountable*: No series of elements of S (even an infinite series) can include all of S's elements. Examples: $\boldsymbol{R}$, $\boldsymbol{R}^2, \wp(N)$

Examples of Countable Sets

- *Theorem*:

  The set of integers is countable.

- *Theorem*:

  The set of all ordered pairs of natural numbers (n,m) is countable.

Example of Uncountable Sets

- *Theorem*:

  The open interval
  $[0,1) = \{r \in \boldsymbol{R} \mid 0 \leq r < 1\}$ is uncountable.

  Proof :

  By *diagonalization*: (Cantor, 1891)

  1. Assume there is a series $\{r_i\} = r_1, r_2, \ldots$ containing *all* elements $r \in [0,1)$.
  2. Consider listing the elements of $\{r_i\}$ in decimal notation (although any base will do) in order of increasing index: ... *(continued on next slide)*

A postulated enumeration of the reals:

$r_1 = 0.d_{1,1}\,d_{1,2}\,d_{1,3}\,d_{1,4}\,d_{1,5}\,d_{1,6}\,d_{1,7}\,d_{1,8}\ldots$

$r_2 = 0.d_{2,1}\,d_{2,2}\,d_{2,3}\,d_{2,4}\,d_{2,5}\,d_{2,6}\,d_{2,7}\,d_{2,8}\ldots$

$r_3 = 0.d_{3,1}\,d_{3,2}\,d_{3,3}\,d_{3,4}\,d_{3,5}\,d_{3,6}\,d_{3,7}\,d_{3,8}\ldots$

$r_4 = 0.d_{4,1}\,d_{4,2}\,d_{4,3}\,d_{4,4}\,d_{4,5}\,d_{4,6}\,d_{4,7}\,d_{4,8}\ldots$

.

.

Now, consider a real number generated by taking all digits $d_{i,i}$ that lie along the *diagonal* in this figure and replacing them with *different* digits.

That real doesn't appear in the list!

Transfinite Numbers

- The cardinalities of infinite sets are not natural numbers, but are special objects called *transfinite* cardinal numbers.
- The cardinality of the natural numbers, $\aleph_0 :\equiv |\boldsymbol{N}|$, is the *first transfinite cardinal* number. (There are none smaller.)
- The *continuum hypothesis* claims that $|\boldsymbol{R}| = \aleph_1$, the *second transfinite cardinal*.
- *Proven impossible to prove or disprove!*

Exercise

1. For each of the following functions, determine
   (1) whether the function is injective, surjective, or bijective
   (2) the image of function
   (3) an express for f^{-1} if the inverse function is defined

(a) $f : \boldsymbol{R} \rightarrow \boldsymbol{R}^{+}, \quad f(x) = 2^{x}$

(b) $f : [0,\infty] \rightarrow \boldsymbol{R}, \quad f(x) = 1/(1+x)$

(c) $f : \boldsymbol{N} \rightarrow \boldsymbol{N}\times\boldsymbol{N}, \quad f(n) = <n, n+1>$

2. Suppose f and $f \circ g$ are one-to-one. Does it follow that g is one to one?

3. Suppose that f is a bijective function from Y to Z and g is a bijective function from X to Y. Show that the inverse $(f \circ g)^{-1}$ of the composition $f \circ g$ given by $(f \circ g)^{-1} = g^{-1} \circ f^{-1}$.

4. Let $f : A \rightarrow B$ and $g : B \rightarrow C$. Prove that

(a) if $f \circ g$ is injective, then f is injective.

(b) if $f \circ g$ is surjective, then g is surjective.

5. Find the cardinal number of each set

(a) A = {a, b, c,, …. ,y, z}.

(b) B = {10, 20, 30, 40, …}.

6. Show that two sets, (-∞,+∞) and (0,1) have the same cardinality.

Discrete Mathematics

5. Graphs & Trees

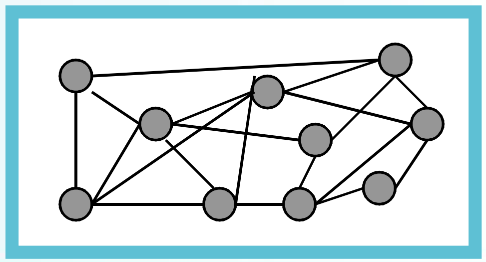
Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

5-1. Graphs

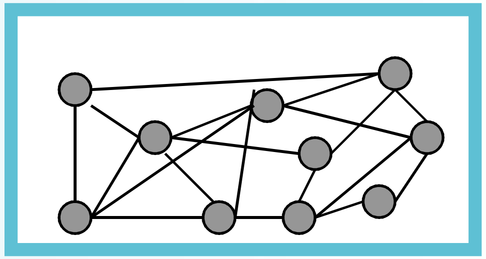
What are Graphs?

- General meaning in everyday math:
  A plot or chart of numerical data using a coordinate system.

- Technical meaning in discrete mathematics:
  A particular class of discrete structures (to be defined) that is useful for representing relations and has a convenient webby-looking graphical representation.



Simple Graphs



Visual Representation of a Simple Graph

- *Definition*:

  A *simple graph* $G=(V,E)$ consists of:

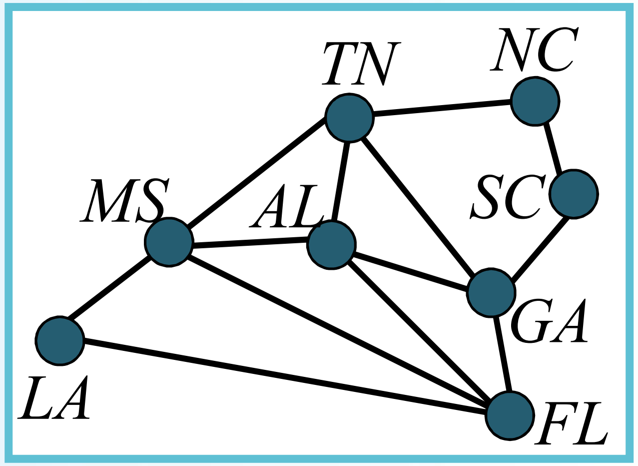
  - a set V of *vertices* or *nodes* (V corresponds to the universe of the relation R), and
  - a set E of *edges* / *arcs* / *links*: unordered pairs of [distinct?] elements $u,v \in V$, such that ${}_uR_v$.

Example of a Simple Graph

- Let V be the set of states in the far-southeastern U.S.:

  V={FL, GA, AL, MS, LA, SC, TN, NC}
- Let E={{u,v}|u adjoins v}

  ={{FL,GA},{FL,AL},{FL,MS},
  {FL,LA},{GA,AL},{AL,MS},
  {MS,LA},{GA,SC},{GA,TN},
  {SC,NC},{NC,TN},{MS,TN},
  {MS,AL}}



Multigraphs

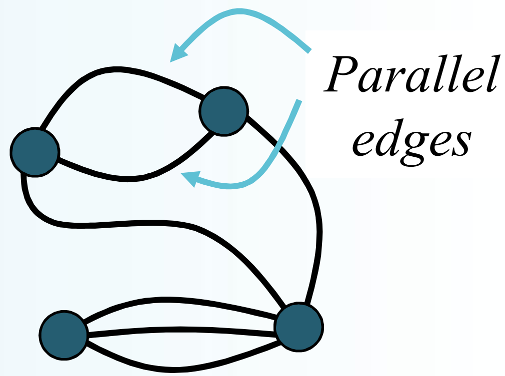
- Like simple graphs, but there may be *more than one* edge connecting two given nodes.

- *Definition*:

  A *multigraph* $G=(V, E, f)$ consists of a set V of vertices, a set E of edges (as primitive objects), and a function $f:E\rightarrow\{\{u,v\}|u,v\in V \wedge u\neq v\}$.

- Example:

  Nodes are cities, edges are segments of major highways.



Pseudographs

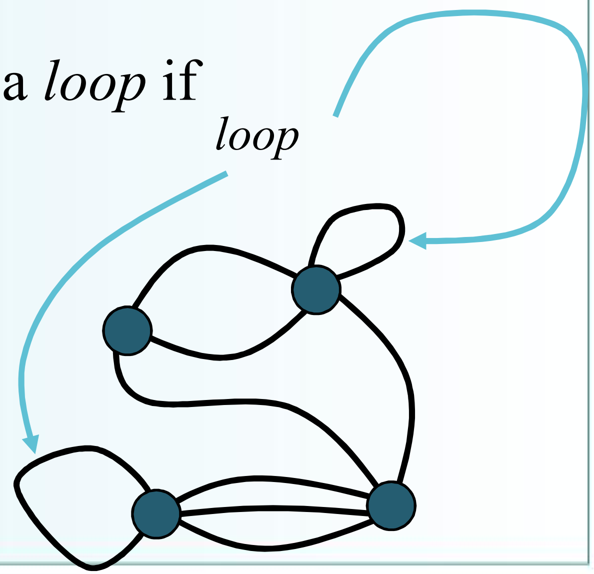
- Like a multigraph, but edges connecting a node to itself are allowed.

- *Definition*:

  A *pseudograph* $G=(V, E, f)$ where $f:E\rightarrow\{\{u,v\}|u,v\in V\}$. Edge $e\in E$ is a *loop* if $f(e)=\{u,u\}=\{u\}$.

- Example:

  Nodes are campsites in a state park, edges are hiking trails through the woods.



Directed Graphs

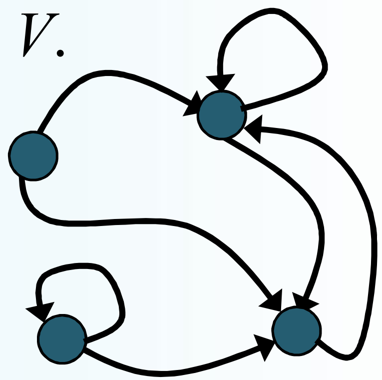
- Correspond to arbitrary binary relations R, which need not be symmetric.

- *Definition*:

  A *directed graph* (V, E) consists of a set of vertices V and a binary relation E on V.

- Example:

  V = people, $E=\{(x,y) \mid x \text{ loves } y\}$



Walk, Path, Cycle, Loop, and Sling

- *Definition*:
  1. A *walk* is a sequence $x_0, x_1, \ldots, x_n$ of the nodes of a digraph such that $x_i x_{i+1}$, $0 \le i \le n-1$, is an edge.
  2. The *length of a walk* is the number of edges in the walk.
  3. A walk $x_0, x_1, \ldots, x_n$ is called a *path* if it holds $x_i \neq x_j$ for $i \neq j$, $i,j = 0, \ldots, n$.
  4. A path $x_0, x_1, \ldots, x_n$ is called a *cycle* if it holds $x_0 = x_n$.
  5. A cycle of length one is called a *loop*.
  6. A cycle of length two is called a *sling*.

Directed Multigraphs

- Like directed graphs, but there may be more than one edge from a node to another.

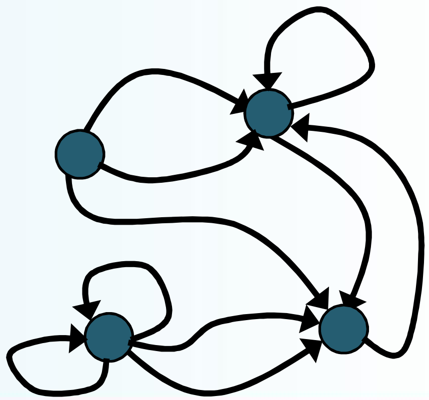
- *Definition*:

  A *directed multigraph* $G=(V, E, f)$ consists of a set V of vertices, a set E of edges, and a function $f:E \rightarrow V \times V$.

- Example:

  The WWW is a directed multigraph.

  - V = web pages, E = hyperlinks.



Types of Graphs: Summary

- Keep in mind this terminology is not fully standardized...

Term	Edge type	Multiple edges ok?	Self-loops ok?
Simple graph	Undir.	No	No
Multigraph	Undir.	Yes	No
Pseudograph	Undir.	Yes	Yes
Directed graph	Directed	No	Yes
Directed multigraph	Directed	Yes	Yes

Graph Terminology

- *Adjacent, connects, endpoints, degree, initial, terminal, in-degree, out-degree, complete, cycles, wheels, n-cubes, bipartite, subgraph, and union.*

Adjacency

Let G be an undirected graph with edge set E.

Let $e \in E$ be (or map to) the pair $\{u,v\}$.

Then we say:

- u, v are *adjacent* / *neighbors* / *connected*.
- Edge e is *incident with* vertices u and v.
- Edge e *connects* u and v.
- Vertices u and v are *endpoints* of edge e.

Degree of a Vertex

- Let G be an undirected graph, $v \in V$ a vertex.
- The *degree* of v, $\deg(v)$, is its number of incident edges. (Except that any self-loops are counted twice.)
- A vertex with degree 0 is *isolated*.
- A vertex of degree 1 is *pendant*.

Handshaking Theorem

- *Theorem*:

  Let G be an undirected (simple, multi-, or pseudo-) graph with vertex set V and edge set E. Then

$$\sum_{v \in V} \deg(v) = 2|E|$$

- *Corollary*:

  Any undirected graph has an even number of vertices of odd degree.

Directed Adjacency

- Let *G* be a directed (possibly multi-) graph, and let *e* be an edge of *G* that is (or maps to) (*u*, *v*). Then we say:
  - *u* is *adjacent to v*, *v* is *adjacent from u*
  - *e comes from* u, e *goes to* v.
  - *e connects u to v*, *e goes from u to v*
  - the *initial vertex* of *e* is *u*
  - the *terminal vertex* of *e* is *v*

Directed Degree

- *Definition*:

  Let G be a directed graph, v a vertex of G.

  1. The *indegree* of v, $\deg^{-}(v)$, is the number of edges going to v.
  2. The *outdegree* of v, $\deg^{+}(v)$, is the number of edges coming from v.
  3. The *degree* of v, $\deg(v)=\deg^{-}(v)+\deg^{+}(v)$, is the sum of v's in-degree and out-degree.

Directed Handshaking Theorem

- *Theorem*:

  Let G be a directed (possibly multi-) graph with vertex set V and edge set E. Then:

  $$\sum_{v \in V} \deg^{-}(v) = \sum_{v \in V} \deg^{+}(v) = \frac{1}{2} \sum_{v \in V} \deg(v) = |E|$$

- Note that the degree of a node is unchanged by whether we consider its edges to be directed or undirected.

Special Graph Structures

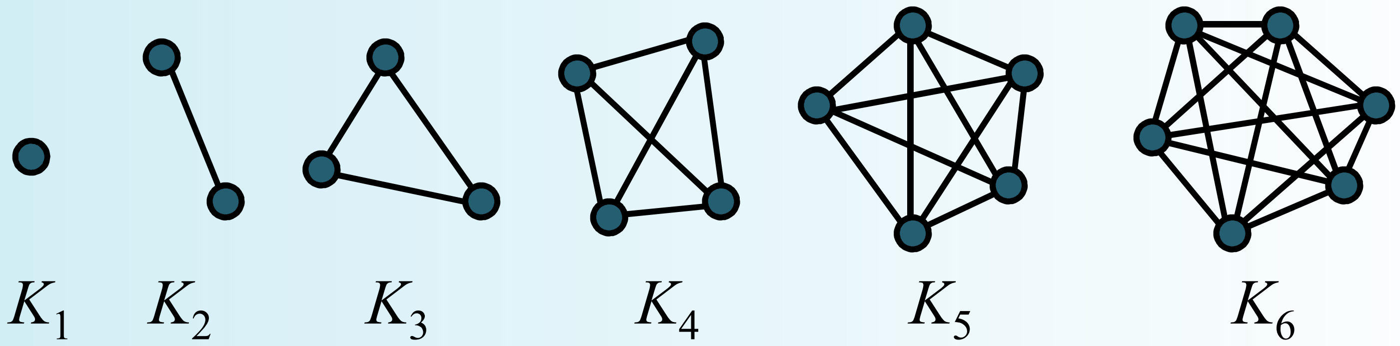
Special cases of undirected graph structures:

- Complete Graphs K_n
- Cycles C_n
- Wheels W_n
- n-Cubes Q_n
- Bipartite Graphs
- Complete Bipartite Graphs $K_{m,n}$

Complete Graphs

- *Definition*:

  For any $n \in \boldsymbol{N}$, a *complete graph* on n vertices, K_n, is a simple graph with n nodes in which every node is adjacent to every other node: $\forall u,v \in V$: $u \neq v \leftrightarrow \{u,v\} \in E$.

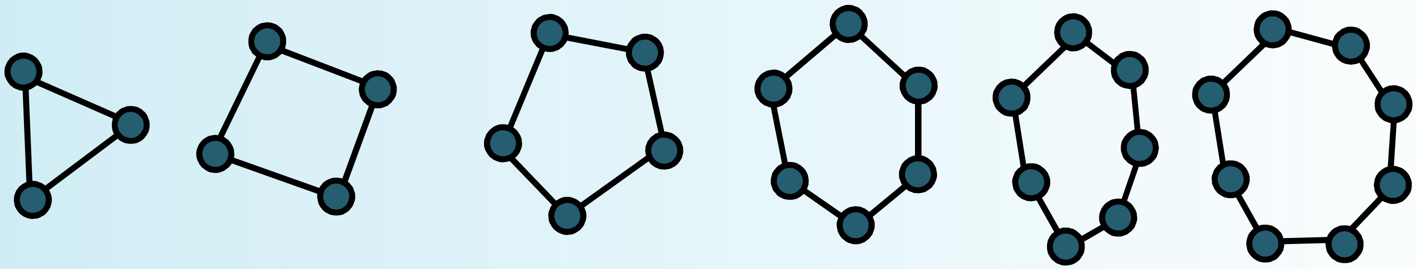


Note that K_n *has* $\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$ *edges.*

Cycles

- *Definition*:

  For any $n \geq 3$, a *cycle* on n vertices, C_n, is a simple graph where $V=\{v_1,v_2,\ldots,v_n\}$ and $E=\{\{v_1,v_2\},\{v_2,v_3\},\ldots,\{v_{n-1},v_n\},\{v_n,v_1\}\}$.



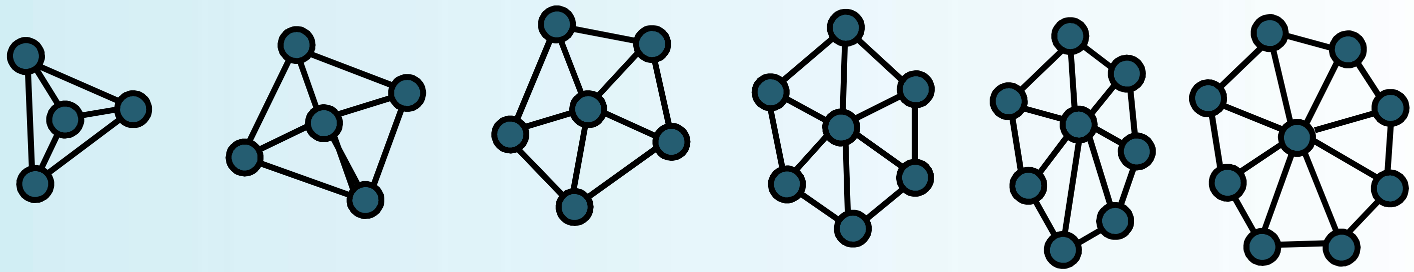
C_3 C_4 C_5 C_6 C_7 C_8

How many edges are there in C_n?

Wheels

- *Definition*:

  For any $n \geq 3$, a *wheel* W_n, is a simple graph obtained by taking the cycle C_n and adding one extra vertex v_{hub} and n extra edges $\{\{v_{\text{hub}}, v_1\}, \{v_{\text{hub}}, v_2\}, \ldots, \{v_{\text{hub}}, v_n\}\}$.



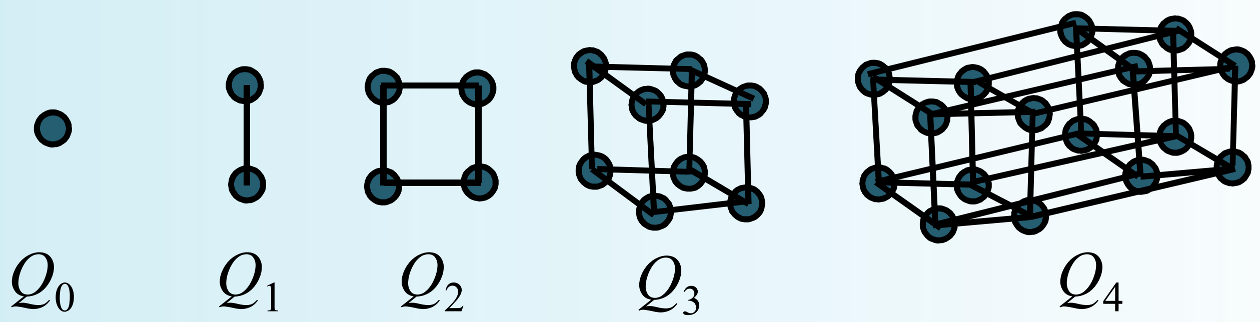
W_3 W_4 W_5 W_6 W_7 W_8

How many edges are there in W_n?

n-Cubes (hypercubes)

- *Definition*:

  For any $n \in \mathbf{N}$, the *hypercube* Q_n is a simple graph consisting of two copies of Q_{n-1} connected together at corresponding nodes. Q_0 has 1 node.



Number of vertices: 2^n. Number of edges:Exercise to try!

- *Definition*:

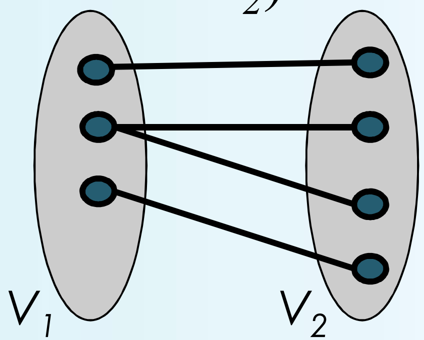
For any $n \in \boldsymbol{N}$, the hypercube Q_n can be defined recursively as follows:

1. $Q_0=\{\{v_0\},\varnothing\}$ (one node and no edges)
2. For any $n \in \boldsymbol{N}$, if $Q_n=(V,E)$, where $V=\{v_1,\ldots,v_a\}$ and $E=\{e_1,\ldots,e_b\}$, then $Q_{n+1}=(V\cup\{v_1',\ldots,v_a'\}, E\cup\{e_1',\ldots,e_b'\}\cup\{\{v_1,v_1'\},\{v_2,v_2'\},\ldots,\{v_a,v_a'\}\})$ where $v_1',\ldots,v_a'$ are new vertices, and where if $e_i=\{v_j,v_k\}$ then $e_i'=\{v_j',v_k'\}$.

Bipartite Graphs

- *Definition*:

  A simple graph G is called *bipartite* if its vertex set V can be partitioned into two disjoint sets V_1 and V_2 such that every edge in the graph connects a vertex in V_1 and a vertex in V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2)



a bipartite

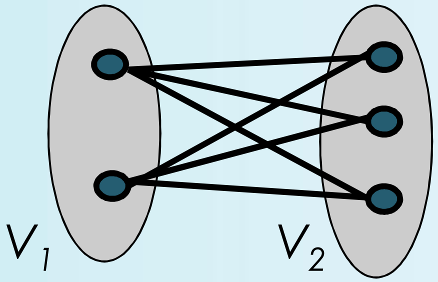
Complete Bipartite Graphs

- *Definition*:

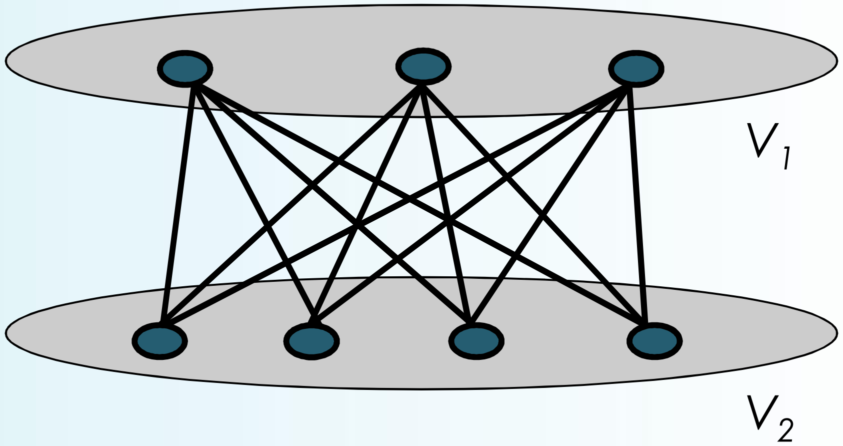
  Let m, n be positive integers. The *complete bipartite graph* $K_{m,n}$ is the graph whose vertices can be partitioned $V = V_1 \cup V_2$ such that

  1. $|V_1| = m$
  2. $|V_2| = n$
  3. For all $x \in V_1$ and for all $y \in V_2$, there is an edge between x and y
  4. No edge has both its endpoints in V_1 or both its endpoints in V_2

Example:



$K_{2,3}$

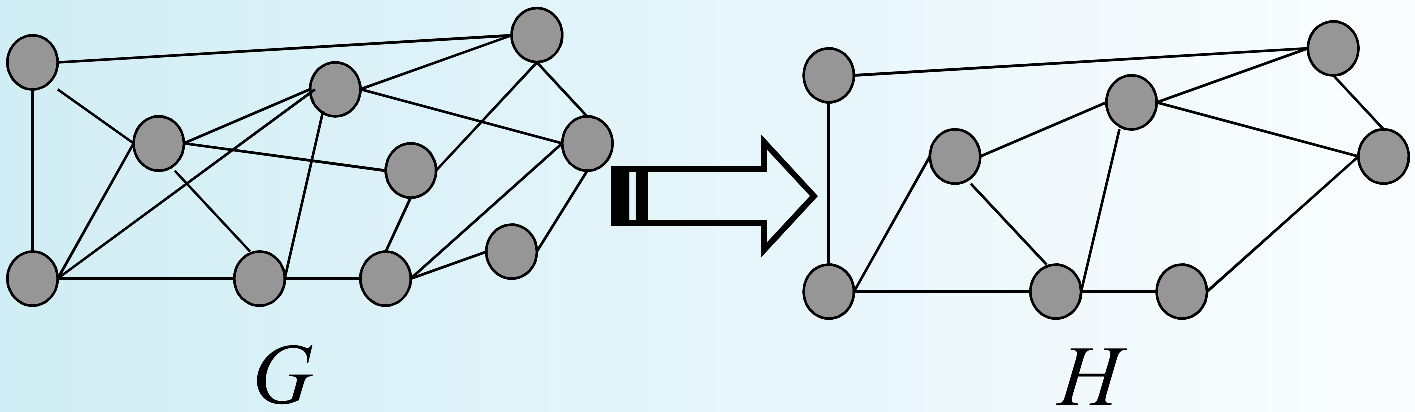


$K_{3,4}$

Subgraphs

- *Definition*:

  A subgraph of a graph $G=(V,E)$ is a graph $H=(W,F)$ where $W \subseteq V$ and $F \subseteq E$.



Graph Unions

- *Definition*:

  The *union* $G_1 \cup G_2$ of two simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2,E_2)$ is the simple graph $(V_1 \cup V_2, E_1 \cup E_2)$.

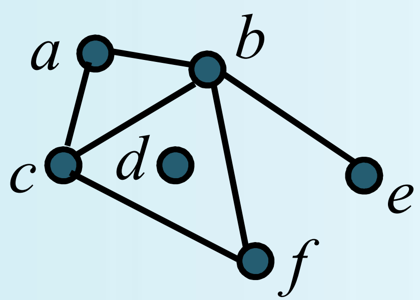
Graph Representations & Isomorphism

- Graph representations:
  - Adjacency lists.
  - Adjacency matrices.
  - Incidence matrices.
- Graph isomorphism:

  Two graphs are isomorphic if and only if they are identical except for their node names.

Adjacency Lists

- A table with 1 row per vertex, listing its adjacent vertices.



Vertex	*Adjacent Vertices*
a	*b, c*
b	*a, c, e, f*
c	*a, b, f*
d	
e	*b*
f	*c, b*

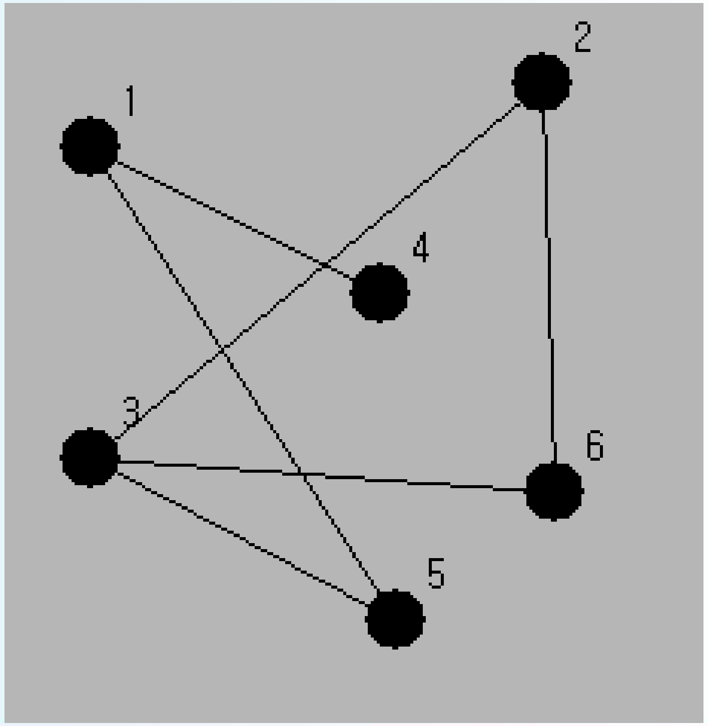
Directed Adjacency Lists

- 1 row per node, listing the terminal nodes of each edge incident from that node.

Adjacency Matrices

- Matrix $\boldsymbol{A}=[a_{ij}]$, where a_{ij} is 1 if $\{v_i, v_j\}$ is an edge of G, 0 otherwise.

$$
\begin{array}{c c}
 & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} &
\begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$



Graph Isomorphism

- *Definition*:

  Simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ are *isomorphic* if there exists a bijection $f:V_1 \rightarrow V_2$ such that for every a and b in V_1, a and b are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2.

- f is the "renaming" function that makes the two graphs identical.
- Definition can easily be extended to other types of graphs.

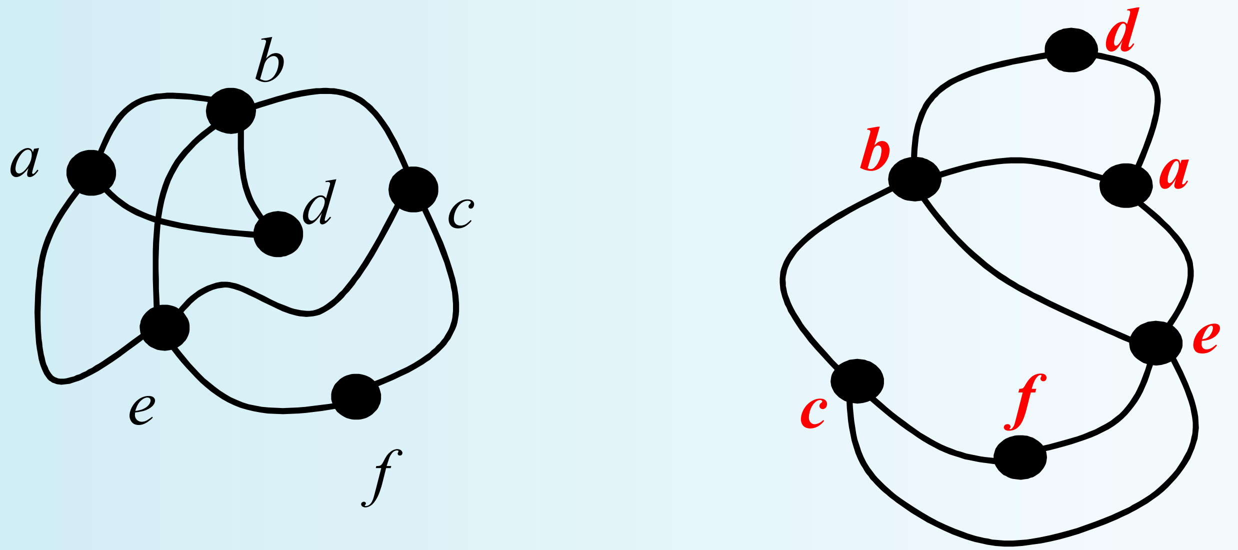
Graph Invariants under Isomorphism

Necessary but not *sufficient* conditions for $G_1=(V_1, E_1)$ to be isomorphic to $G_2=(V_2, E_2)$:

1. $|V_1|=|V_2|$ and $|E_1|=|E_2|$.
2. The number of vertices with degree n is the same in both graphs.
3. For every proper subgraph g of one graph, there is a proper subgraph of the other graph that is isomorphic to g.

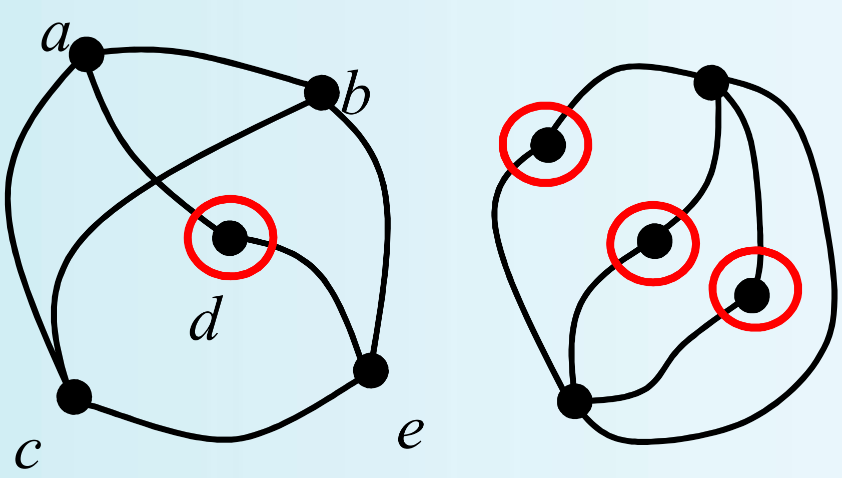
Isomorphism Example

- If isomorphic, label the *2*nd graph to show the isomorphism, else identify difference.



Are these Isomorphic?

- If isomorphic, label the *2*nd graph to show the isomorphism, else identify difference.



** Same # of vertices*

** Same # of edges*

** Different # of verts of degree 2! (1 vs. 3)*

Connectedness

- *Definition*:

  An undirected graph is *connected* if and only if there is a walk between every pair of distinct vertices in the graph.

- *Theorem*:

  There is a path between any pair of vertices in a connected undirected graph.

Directed Connectedness

- *Definition*:
  1. A directed graph is *strongly connected* if there is a directed path from *a* to *b* for any two vertices *a* and *b*.
  2. It is *weakly connected* if the underlying *undirected* graph (*i.e.*, with edge directions removed) is connected.

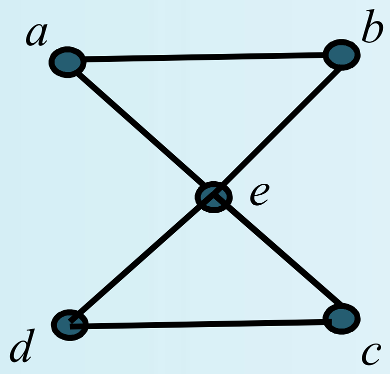
- Note that *strongly* implies *weakly* but not vice-versa.

Euler Circuits and Paths

- *Definition*:

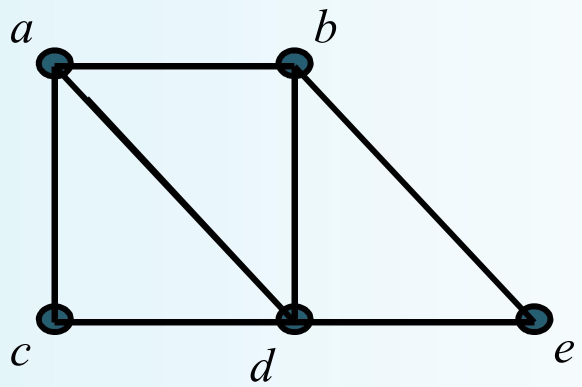
1. An *Euler circuit* in a graph G is a circuit containing every edge of G.
2. An *Euler path* in G is a walk containing every edge of G.

- Example:



a, e, c, d, e, b, a

Euler circuit



a, c, d, e, b, d, a, b

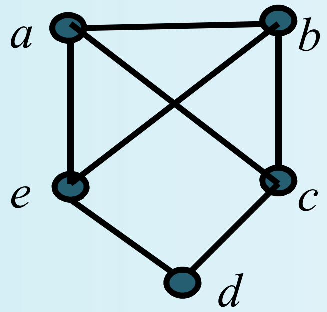
Euler path

Hamilton Circuits and Paths

- *Definition*:

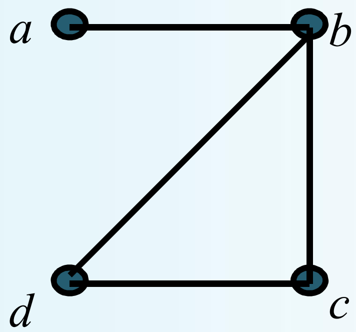
1. A *Hamilton circuit* is a circuit that traverses each vertex in *G* exactly once.
2. A *Hamilton path* is a walk that traverses each vertex in G exactly once.

- Example:



a, b, c, d, e, a

Hamilton circuit



a, b, c, d

Hamilton path

5-2. Trees

Trees

- *Definition*:

  A *tree* is an acyclic directed graph such that (1) there is exactly one node, called the root of the tree, which has indegree 0, (2) every node other than the root has indegree l, and (3) for every node *a* of the tree, there is a directed path from the root to *a*.

- *Definition:*

  In a tree, any node which has outdegree 0 is called a *terminal node* or a *leaf*; all other nodes are called *branch/ interior/ internal nodes*. The *level* of any node is the length of its path from the root where the level of the root is 0. The *height* of the tree is the maximum of the levels of nodes.

- *Definition*:

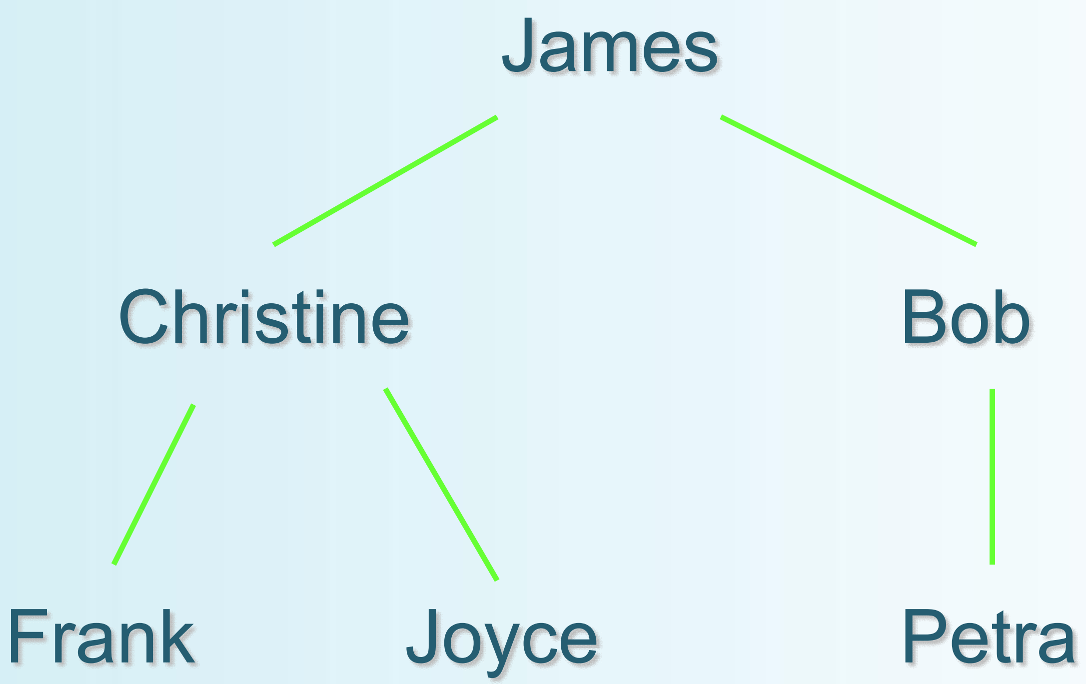
1. If v is a node in a tree other than the root, the *parent* of v is the unique node u such that there is a directed edge from u to v.
2. When u is the parent of v, v is called the *child* of u.
3. Nodes with the same parent are called *siblings*.
4. The *ancestors* of a node other than the root are those nodes in the path from the root to this node, excluding the node itself but including the root.
5. The *descendants* of a node v are those nodes that have v as an ancestor.

- *Definition*:

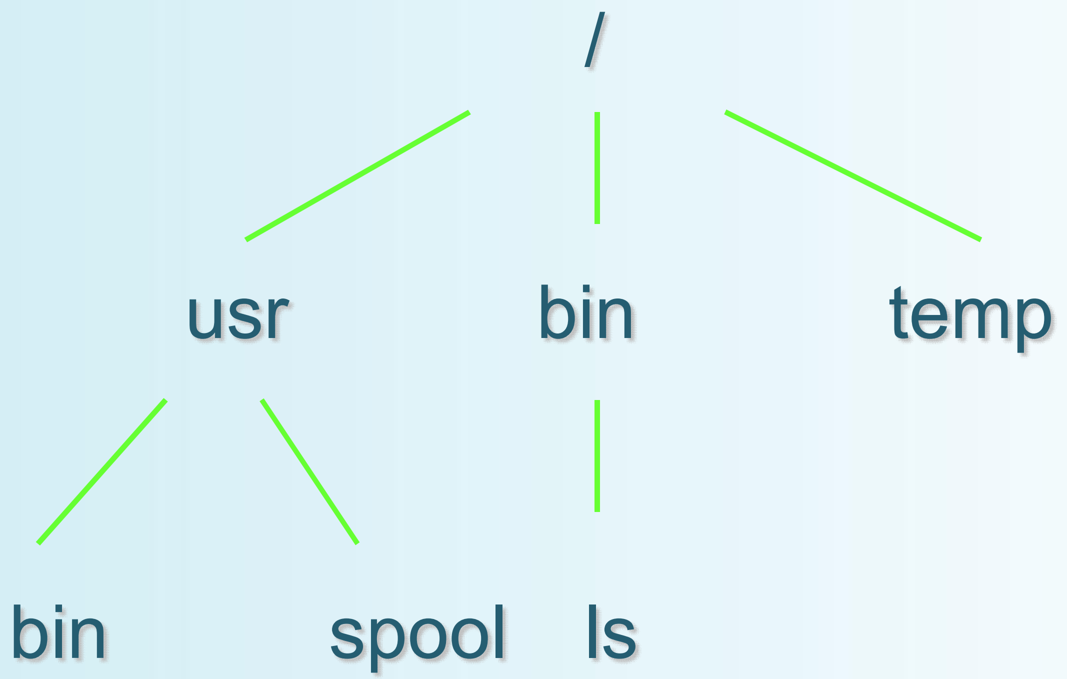
  If *a* is a node in a tree, then the *subtree* with *a* as its root is the subgraph of the tree consisting of *a* and its descendants.

Examples

Example 1: Family tree



Example 2: File system



Example 3: Arithmetic expressions

- This tree represents the expression (y + z)·(x - y).

- *Definition*:

1. A tree is called an *m-ary tree* if every internal vertex has no more than *m* children.
2. A tree is called a *full m-ary tree* if every internal vertex has exactly *m* children.
3. An *m*-ary tree with *m* = 2 is called a *binary tree.*

- *Theorem*:

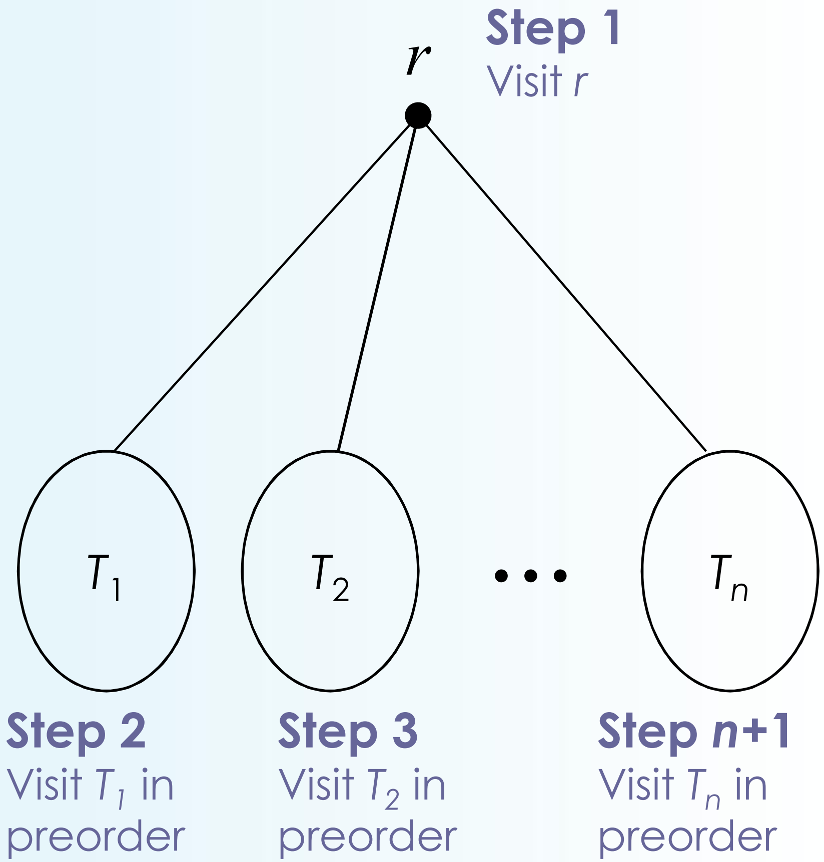
1. A tree with n vertices has $(n - 1)$ edges.
2. A full m-ary tree with i internal vertices contains $n = m \cdot i + 1$ vertices.

Tree Traversal

- Procedures for systematically visiting every vertex of an ordered tree are called traversal algorithms.

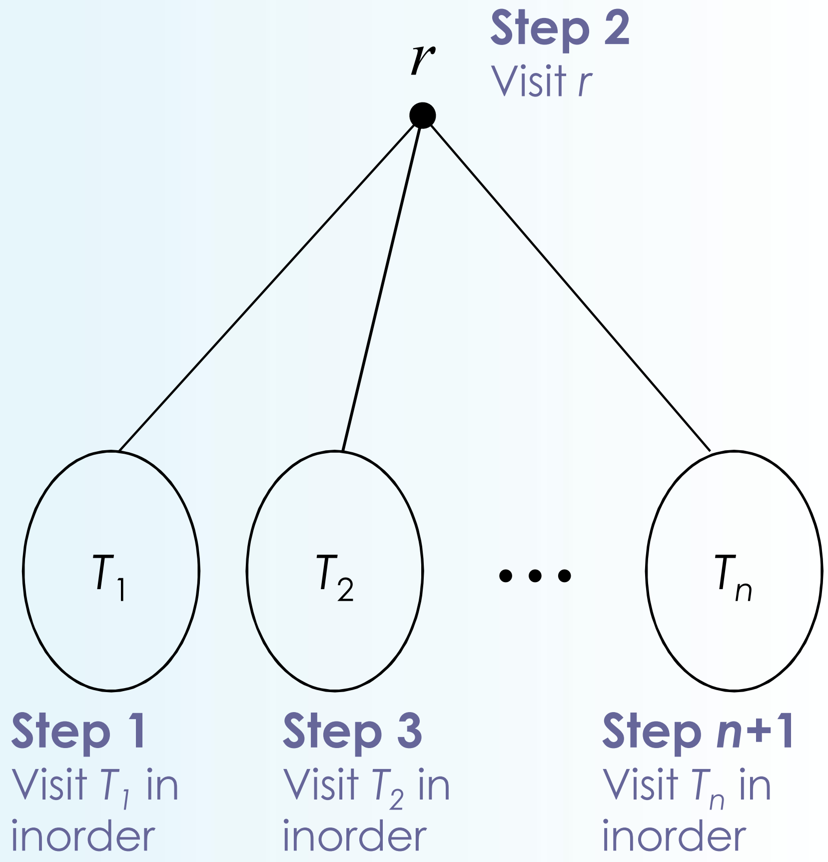
Preorder Traversal

- *Definition*:
  Let T be an ordered tree with root r.
  If T consists only of r, then r is the *preorder traversal* of T.
  Otherwise, suppose that T_1, T_2, …, T_n are the subtrees at r from left to right in T. The *preorder traversal* begins by visiting r. It continues by traversing T_1 in preorder, then T_2 in preorder, and so on, until T_n is traversed in preorder.



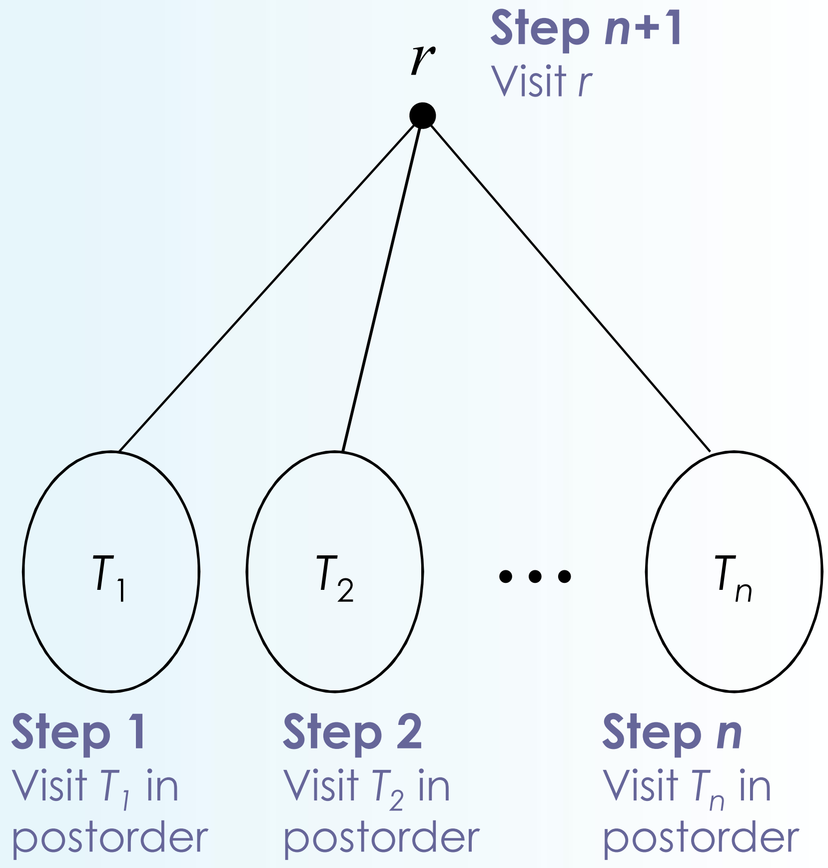
Inorder Traversal

- *Definition*:
  Let T be an ordered tree with root r.
  If T consists only of r, then r is the *inorder traversal* of T.
  Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at r from left to right. The *inorder traversal* begins by traversing visiting T_1 in inorder, then visiting r. It continues by traversing T_2 in inorder, then T_3 in inorder, …, and finally T_n in inorder

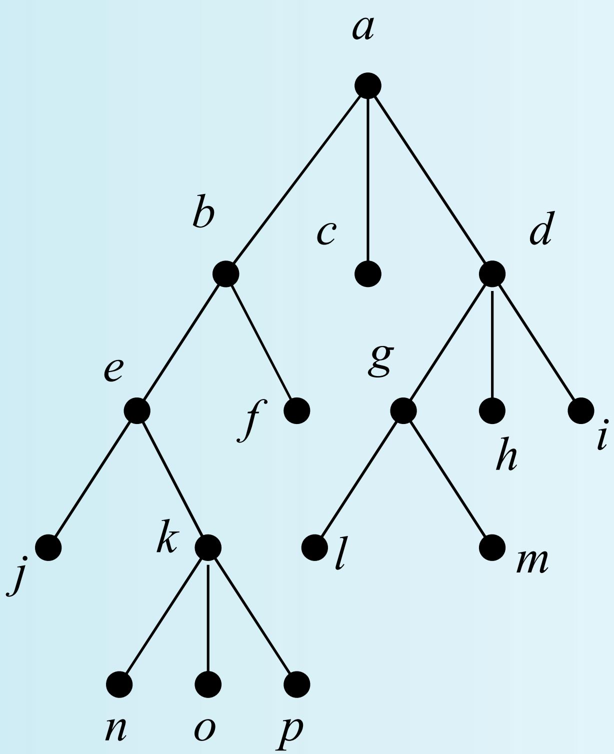


Postorder Traversal

- *Definition*:
  Let T be an ordered tree with root r. If T consists only of r, then r is the *postorder traversal* of T. Otherwise, suppose that T_1, T_2, …, T_n are the subtrees at r from left to right. The *postorder traversal* begins by traversing T_1 in postorder, then T_2 in postorder, …, then T_n in postorder, and ends by visiting r.



Example of Traversal



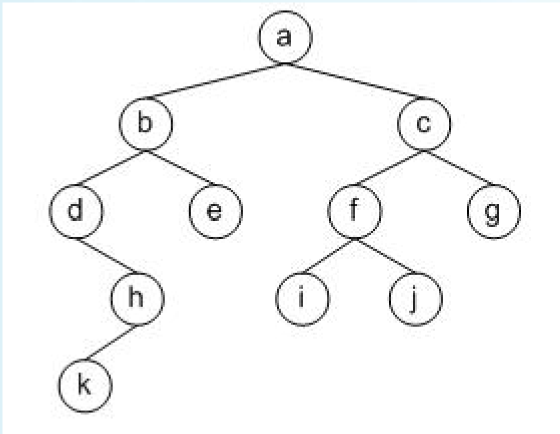
- *Preorder* : *a, b, e, j, k, n, o, p, f, c, d, g, l, m, h, i*
- *Inorder* : *j, e, n, k, o, p, b, f, a, c, l, g, m, d, h, i*
- *Postorder* : *j, n, o, p, k, e, f, b, c, l, m, g, h, i, d, a*

Exercise

1. Let G be a graph. Prove that there must be an even number of vertices of odd degree.

2. Prove that in any graph with two or more vertices, there must be two vertices of the same degree.

3. List the order of the nodes of the following binary tree visited by each of *preorder*, *inorder*, and *postorder* traversal algorithm.



Discrete Mathematics

6. Algebras, Lattices & Boolean Functions

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

6-1. Algebras

Algebra

- *Definition*:

  An *algebra* is characterized by specifying the following three components:

  - A set called the *carrier* of the algebra,
  - *Operators* defined on the carrier, and
  - Distinguished elements of the carrier, called the *constants* of the algebra.

Closed with respect to operation

- *Definition*:

  Let $\circ$ and Δ be binary and unary operations on a set T and let T' be a subset of T. Then T' is *closed with respect to* $\circ$, if $a, b \in T'$ implies $a \circ b \in T'$. The subset T' is *closed with respect to* Δ, if $a \in T'$ implies $\Delta a \in T'$.

Subalgebra

- *Definition*:

  Let $A=<S, \circ, \Delta, k>$ and $A'=<S', \circ', \Delta', k'>$ be algebras. Then A' is a *subalgebra* of A if

  - $S' \subseteq S$
  - $a \circ' b = a \circ b$ for all $a, b \in S'$
  - $\Delta' a = \Delta a$ for all $a \in S'$
  - $k' = k$.

Identity and Zero Element

- *Definition*:

  Let $\circ$ be a binary operation of S.

  – An element $1 \in S$ is an *identity* (or *unit*) for the operation $\circ$ if every $x \in S$,

  $$1 \circ x = x \circ 1 = x$$

  – An element $0 \in S$ is a *zero* for the operation $\circ$ if for every $x \in S$

  $$0 \circ x = x \circ 0 = 0$$

- *Definition*:

  Let $\circ$ be a binary operation on S.

  1. An element 1_l (1_r) is a *left (right) identity for the operation* $\circ$ if for every $x \in S$,

     $$1_l \circ x = x \quad (x \circ 1_r = x)$$

  2. An element 0_l (0_r) is a *left (right) zero for the operation* $\circ$. If for every $x \in S$,

     $$0_l \circ x = 0 \quad (x \circ 0_r = 0)$$

Inverse Element

- *Definition*:

  Let $\circ$ be a binary operation on S and 1 an identity for the operation $\circ$.

  1. If $x \circ y=1$, then x is a *left inverse* of y and y is a *right inverse* of x with respect to the operation $\circ$.
  2. If both $x \circ y=1$ and $y \circ x=1$ then x is an *inverse* of y with respect to the operation $\circ$.

Semigroup

- *Definition*:

  A *semigroup* is an algebra with signature <*S*, ∘> where ∘ is a binary associative operation: for every *a*, *b*, and *c* in *S*, $a \circ (b \circ c) = (a \circ b) \circ c$

- *Theorem*:

  If <*S*, ∘> is a semigroup and <*T*, ∘> is a subalgebra of <*S*, ∘>, the <*T*, ∘> is a semigroup.

Monoid

- *Definition*:

  A *monoid* is an algebra with signature $<S, \circ, 1>$ where $\circ$ is a binary associative operation on S and 1 is an identity for the operation $\circ$. i.e. the following axioms hold for all elements a, b, and c in S:

  - $a\circ(b\circ c) = (a\circ b)\circ c$
  - $a\circ 1 = a$
  - $1\circ a = a$

Group

- *Definition*:

  A *group* is an algebra with signature $<S, \circ, \overline{\ }, 1>$ where $\circ$ is an associative binary operation on S, the constant 1 is an identity for the operation on $\circ$ and $\overline{\ }$ is a unary operation defined over S such that for all $x \in S$, $\overline{x}$ is an inverse for x with respect to $\circ$.

- *Theorem:*

  Let $<S, \circ, \overline{\ }, 1>$ be a group. Every element of S has a unique inverse in S.

Homomorphism

- *Definition*:

  Let A=<S, $\circ$, Δ, k> and A'=<S', $\circ'$, Δ', k'> be two algebras with the same signature and let the function $h{:}S \rightarrow S'$ be such that

  - $h(x \circ y) = h(x) \circ' h(y)$,
  - $h(\Delta x) = \Delta' h(x)$
  - $h(k) = k'$.

  Then h is called *homomorphism* for A to A'.

Epimorphism, Monomorphism, and Isomorphism

- *Definition*:
  1. h is *epimorphism* if h is onto and homomorphism.
  2. h is *monomorphism* if h is one-to-one and homomorphism.
  3. h is *isomorphism* if h is bijection and homomorphism.

Congruence Relation

- *Definition*:

  Given an algebra $A=<S, \circ, \Delta>$ with a binary operation $\circ$ and a unary operation Δ, an equivalence relation E on S is a *right* (*left*) *congruence relation* on A if and only if for every x, y, and z in S,

1. if $<x, y> \in E$, then $<x \circ z, y \circ z> \in E$ $(<z \circ x, z \circ y> \in E)$
2. if $<x, y> \in E$, then $<\Delta x, \Delta y> \in E$.

- *Definition*:

  Given an algebra $A=<S, \circ, \Delta>$, an equivalence relation E on S is a congruence relation on A if and only if it is a *left and right congruence relation on A*.

- *Theorem:*

  Let $A=<S, \circ>$ be an algebra with a binary operation $\circ$ and let E be an equivalence relation on S. Then E is a congruence relation on A if and only if for every $x_1, x_2, y_1,$ and y_2 in S,

  $(<x_1, x_2> \in E \wedge <y_1, y_2> \in E) \Rightarrow <x_1 \circ y_1, x_2 \circ y_2> \in E$

6-2. Lattices

Lattices

- *Definition*:

  A poset $<L,\leq>$ is a *lattice* if every two elements in L has a *lub* and a *glb*.

- *Theorem*:

  Let $<L, \leq>$ be a lattice. Then for every a, b, and c in L,

  1. $a*a=a,\ a+a=a$ (*idempotent*)
  2. $a*b=b*a,\ a+b=b+a$ (*commutative*)
  3. $(a*b)*c= a*(b*c),\ (a+b)+c= a+(b+c)$ (*associative*)
  4. $a*(a+b)=a,\ a+(a*b)=a$ *(absorption)*

  where * and + represent the *glb* and the *lub*, respectively.

- *Theorem*:

  Let $<L, \leq>$ be a lattice. Then for every a and b in L,
  $a \leq b$ if and only if $a*b=a \Leftrightarrow a+b=b$

- *Theorem*:

  Let $<L, \leq>$ be a lattice. Then for every a, b, and c in L,
  if $b \leq c$, then $a*b \leq a*c$ and $a+b \leq a+c$

- *Theorem*:

  Let $< L, \leq>$ be a lattice. Then for every a, b, and c in L,
  $a+(b*c) \leq (a+b)*(a+c)$ and $(a*b)+(a*c) \leq a*(b+c)$

- *Theorem*:

  Let $<A, *, +>$ be an algebra with two binary operations $*$ and $+$. If the following property holds that for any a, b, and c in A,

  1. $a*a=a, \ a+a=a$ (*idempotent*)
  2. $a*b=b*a, \ a+b=b+a$ (*commutative*)
  3. $(a*b)*c= a*(b*c), \ (a+b)+c= a+(b+c)$ (*associative*)
  4. $a*(a+b)=a, \ a+(a*b)=a$ (*absorption*),

  then there exists a lattice $<A, \leq>$ such that $*$ is a *glb*, $+$ is a *lub*, and $\leq$ is defined as $x\leq y$ iff $x*y=x$ $(x+y=y)$.

- *Definition*:

  A *lattice* is an algebraic system $<L, *, +>$ with two binary operations * and + on L which are both *commutative* and *associative* and satisfy the *absorption* law.

- *Definition*:

  Let $<L, *, +>$ be a lattice and let $S \subseteq L$ be a subset of L. The algebra $<S, *, +>$ is a *sublattice* of $<L, *, +>$ if S is closed under both operations * and +.

- *Definition*:

  Let $<L, *, +>$ and $<S, \cap, \cup>$ be two lattices.

  A mapping g:$L \rightarrow S$ is called a *lattice homomorphism* from the lattice $<L, *, +>$ to $<S, \cap, \cup>$ if for any a and b in L,

  $g(a*b)=g(a) \cap g(b)$ and $g(a+b)=g(a) \cup g(b)$.

- *Definition*:

  Let $<P, \leq>$ and $<Q, \leq'>$ be two partially ordered sets,

  A mapping f:$P \rightarrow Q$ is said to be *order-preserving* relative to the ordering $\leq$ in P and $\leq'$ in Q if for every a and b in P,

  $a \leq b$ implies $f(a) \leq' f(b)$ in Q.

- *Definition*:

  Two partially ordered sets $<P, \leq>$ and $<Q, \leq'>$ are called *order-isomorphic* if there exists a bijection $f{:}P \rightarrow Q$ and if both f and f^{-1} are order-preserving.

- *Definition*:

  A lattice is called *complete* if each of its nonempty subsets has a *lub* and a *glb*.

- *Definition*:

  The least and the greatest elements of a lattice, if they exist, are called the *bounds* of the lattice, and are denoted by 0 and 1 respectively.

- *Definition*:

  In a bounded lattice $<L, *, +, 0, 1>$, an element b in L is called a *complement* of an element a in L if $a*b=0$ and $a+b=1$.

- *Theorem*:

  In a bounded lattice $<L, *, +, 0, 1>$, $1(0)$ is the only complement of $0(1)$.

- *Definition*:

  A lattice $<L, *, +, 0, 1>$ is said to be a *complemented lattice* if every element in L has at least one complement.

- *Definition*:

  A lattice $<L, *, +>$ is called a *distributive lattice* if for every a, b, and c in L,

  $$a*(b+c)=(a*b)+(a*c) \quad \text{and} \quad a+(b*c)=(a+b)*(a+c)$$

- *Theorem*:

  Every chain is a distributive lattice.

Exercise

1. Let the algebra, A = < I, + >, where I is a set of integers and + is a binary addition operation. For each of the following binary relations defined on I, prove or disprove that the relation is a congruence relation on A.
   (a) $< x, y > \in R_1$ if and only if $|x - y|<10$
   (b) $< x, y > \in R_2$ if and only if $x \geq y$
   (c) $< x, y > \in R_3$ if and only if $(x <0 \wedge y <0) \vee (x \geq 0 \wedge y \geq 0)$

2. Let A = $<S, +>$ and B=$<T, \cdot>$ be two algebras with binary operations + and $\cdot$, and let the function, $h{:}S \rightarrow T$, be a homomorphism from A to B. Show that the relation R on S defined to be $< x, y> \in R$ iff $h(x)=h(y)$ is a congruence relation on A.

3. Let <$\boldsymbol{R}$,+,0> and <$\boldsymbol{R}$,·,1> be two algebra where $\boldsymbol{R}$ is a set of reals, + is a binary addition, and · is a binary multiplication. When the function, f:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, is defined to be $f(x) = 2^x$, answer the following with justification:

Is f homomorphism from <$\boldsymbol{R}$,+,0> to <$\boldsymbol{R}$,·,1>?

6-3. Boolean Functions

Boolean Lattice & Boolean Algebra

- *Definition*:

1. A *Boolean lattice* is a complemented and distributive lattice.
2. A *Boolean algebra* is an algebra with signature $<B, +, *, ', 0, 1>$, where + and * are binary operations and ′ is a unary operation called complementation, and the following axioms hold:
   - $x*x=x,\ x+x=x$ (*idempotent*)
   - $(x*y)*z=x*(y*z),\ (x+y)+z=x+(y+z)$ (*associative*)
   - $x*y=y*x,\ x+y=y+x$ (*commutative*)
   - $x*(x+y)=x,\ x+(x*y)=x$ (*absorption*)
   - $x*(y+z)=(x*y)+(x*z),\ x+(y*z)=(x+y)*(x+z)$ (*distributive*)
   - Every element x has a (unique) complement x' such that $x*x'=0$ and $x+x'=1$ (*complemented*).

- *Theorem*:

  Let $<B, *, +, ', 0, 1>$ be a *Boolean algebra*. Then $<B, \leq>$ is a *Boolean lattice* when the relation $\leq$ is defined to be $x \leq y$ if and only if $x*y=x$ $(x+y=y)$ for x, y in B.

  Proof:

  1. Show that $\leq$ is *a partial ordering*.
  2. Show that $(x*y)$ and $(x+y)$ represent the *glb* and the *lub* of x and y, respectively.

- *Theorem* (*Stone's Representation Theorem*):

  For every *Boolean algebra* <B, *, +, ', 0, 1>, there exists a power set algebra <$\wp(A)$, $\cap$, $\cup$, $\overline{}$, $\emptyset$, *A*> which is isomorphic to <*B*, *, +, ', 0, 1>.

Proof:

Given a *Boolean algebra* <*B*, *, +, ' , 0, 1>,

1. define an *atom* to be the element in *B* that *covers* 0 (for x and y in *B*, x covers y iff $y \le x$ and there is no z in *B* such that $y \le z$ and $z \le x$),

2. define $f : B \rightarrow \wp(A)$, where A is a set of atoms, such that for any x in B, $f(x) = \{ a \mid (a \in A)$ and $(a \leq x) \}$, and

3. show that f is isomorphism from $\langle B, *, +, ', 0, 1 \rangle$ to $\langle \wp(A), \cap, \cup, \overline{\ \ }, \varnothing, A \rangle$.

Lemma 1:
For every $x \neq 0$ in B, $\exists\, a \in A$, such that $a \leq x$

Lemma 2:
For every $x \neq 0$ in B and a in A, one and only one of the following holds.

1. $a \leq x$
2. $a * x = 0$ $(a \leq x')$

Lemma 3: (*homomorphism*)

$$f(x') = \overline{f(x)}$$

Lemma 4: (*homomorphism*)

1. $f(x*y) = f(x) \cap f(y)$
2. $f(x+y) = f(x) \cup f(y)$

Lemma 5: (*one-to-one*)
$x = y$ if $f(x) = f(y)$

Lemma 6: (*onto*)
For any $\{a_1, a_2, \ldots, a_k\} \subseteq A$,
$\exists\ (a_1 + a_2 + \ldots + a_k) \in B$ such that $f(a_1 + a_2 + \ldots + a_k) = \{a_1, a_2, \ldots, a_k\}$.

Boolean Expression

- *Definition :*

  A *Boolean expression* in *n* variables, $x_1, x_2, \ldots, x_n$, is a finite string of symbols formed by the following:

  1. 0 and 1 are *Boolean expressions*.
  2. $x_1, x_2, \ldots, x_n$ are *Boolean expressions*.
  3. If *p* and *q* are *Boolean expressions* the $(p*q)$ and $(p+q)$ are *Boolean expressions.*
  4. If *p* is a *Boolean expression*, then p' is a *Boolean expression*.
  5. No string of symbols except those formed by steps 1, 2, 3, and 4 is a *Boolean expression*.

Equivalence

- *Definition*:

  Two *Boolean expressions,* $\alpha\ (x_1, x_2, \ldots, x_n)$ and $\beta\ (x_1, x_2, \ldots, x_n)$, are *equivalent* if one can be obtained from the other by a finite number of applications of identities of a *Boolean algebra*.

- *Definition*:

  Let $\alpha\ (x_1, x_2, \ldots, x_n)$ be a *Boolean expression* in *n* variables and $<B, *, +, ', 0,1>$ be any *Boolean algebra* whose elements are denoted by $a_1, a_2, \ldots, a_n$. Let $<a_1, a_2, \ldots, a_n>$ be an *n*-tuple of B^n. Then the *value* of the *Boolean expression* $\alpha\ (x_1, x_2, \ldots, x_n)$ for the *n*-tuple $<a_1, a_2, \ldots, a_n> \in B^n$ is given by $\alpha\ (a_1, a_2, \ldots, a_n)$ which is obtained by replacing x_1 by a_1, x_2 by a_2, ..., and x_n by a_n in the $\alpha\ (x_1, x_2, \ldots, x_n)$.

Boolean Function

- *Definition*:

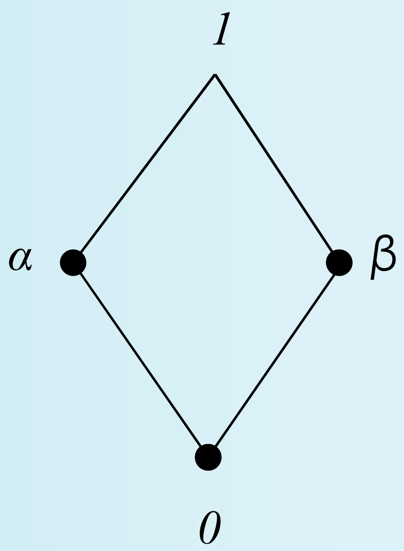
  Let f:$B^n \rightarrow B$ be a function. If a *Boolean expression* $g(x_1, x_2, \ldots, x_n)$ matches to a function f, then we say g is *associated with* function f.

- *Definition*:

  Let <B, *, +, ', 0, 1> be a *Boolean algebra*. A function f:$B^n \rightarrow B$ which is associated with a *Boolean expression* in n variables is called a *Boolean function*. A *Boolean function* defined on a switching algebra is called a *switching function*.

Example

- Which of f_1, f_2, and f_3 are *Boolean functions* ? (f_i: $B^2 \rightarrow B$, i=1,2,3)



$<B, *, +, ', 0, 1>$

where $B = \{ 0, 1, \alpha, \beta \}$

$f_1 = x_1'x_2 + x_1x_2'$

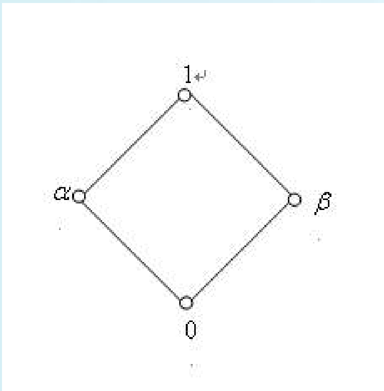
x_1, x_2	f_1	f_2	f_3
$0, 0$	0	1	0
$0, \alpha$	α	β	β
$0, \beta$	β	α	β
$0, 1$	1	0	α
$\alpha, 0$	α	β	0
α, α	0	β	1
α, β	1	0	α
$\alpha, 1$	β	0	0
$\beta, 0$	β	β	α
β, α	1	0	0
β, β	0	α	β
$\beta, 1$	α	β	α
$1, 0$	1	0	β
$1, \alpha$	β	α	α
$1, \beta$	α	β	β
$1, 1$	0	0	1

Exercise

1. Let $<B, \leq_1>$ be a *Boolean lattice* where $B=\{1,2,3,5,6,10,15,30\}$ and $\leq_1$ is defined to be "$x \leq_1 y$ if and only if x divides y".
   By *Stone Representation Theorem*, there exists a power set *Boolean lattice*, $<\wp(A), \leq_2>$, which is isomorphic to $<B, \leq_1>$.
   Answer each of the following:
   (a) Define set A.
   (b) Show that $f{:}B\rightarrow\wp(A)$ is a homomorphism from $<B, \leq_1>$ to $<\wp(A), \leq_2>$.

2. Let $<B, +, *, ', 0, 1>$ be a *Boolean algebra*. Show that the complement x' of each element x in B is unique .

3. Let the *Boolean algebra* < B, *, +, ′, 0, 1> have the following Hasse diagram. For each of three functions f_1, f_2, and f_3 given in the table, indicate whether or not it is a *Boolean function*. If it is, give the corresponding *Boolean expression* in two variables, x_1 and x_2.



x_1	x_2	f_1	f_2	f_3
0	0	0	1	1
0	α	α	α	β
0	β	β	β	α
0	1	1	1	0
α	0	0	β	1
α	α	0	1	1
α	β	β	β	α
α	1	β	α	α
β	0	0	α	1
β	α	α	α	β
β	β	0	1	1
β	1	α	1	β
1	0	0	0	1
1	α	0	α	1
1	β	0	β	1
1	1	0	1	1

Discrete Mathematics

7. Algorithms and Complexity

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

7-1. Algorithms

Algorithms

- The foundation of computer programming.
- Most generally, an *algorithm* just means a definite procedure for performing some sort of task.
- A computer *program* is simply a description of an algorithm in a language precise enough for a computer to understand, requiring only operations the computer already knows how to do.
- We say that a program *implements* its algorithm.

Programming Languages

- Some common programming languages:
  - Newer: Java, C, C++, Visual Basic, JavaScript, Perl, Tcl, Pascal
  - Older: Fortran, Cobol, Lisp, Basic
  - Assembly languages, for low-level coding.
- In this class we will use an informal, Pascal-like "*pseudo-code*" language.

Example of Algorithm

- Task: Given a sequence $\{a_i\}=a_1,\ldots,a_n$, $a_i \in \mathbf{N}$, say what its largest element is.
- Set the value of a *temporary variable* v (largest element seen so far) to a_1's value.
- Look at the next element a_i in the sequence.
- If $a_i > v$, then re-assign v to the number a_i.
- Repeat previous 2 steps until there are no more elements in the sequence, & return v.

Executing an Algorithm

- When you start up a piece of software, we say the program or its algorithm are being *run* or *executed* by the computer.
- Given a description of an algorithm, you can also execute it by hand, by working through all of its steps on paper.

Executing the MAX algorithm

1. Let $\{a_i\}$=7,12,3,15,8.   Find its maximum…
2. Set $v = a_1 = 7$.
3. Look at next element: $a_2 = 12$.
4. Is $a_2 > v$? Yes, so change v to 12.
5. Look at next element: $a_2 = 3$.
6. Is 3>12? No, leave v alone….
7. Is 15>12? Yes, v=15…

Examples:

Algorithm (Procedure) MAX(a_1, a_2, …, a_n)

begin

$max := a_1$

for $i := 2$ to n

if $max < a_i$ **then** $max := a_i$

{ max is the largest element }

end

Algorithm *Linear Search* ($x, a_1, a_2, \ldots, a_n$)

begin

$i := 1$

while ($i \leq n$ and $x \neq a_i$)

$i := i+1$

if $i \leq n$ **then** *location* $:= i$

else *location* $:= 0$

{ *location* is the subscript of the term that equals x, or is 0 if x is not found }

end

Algorithm *BinarySearch* (x, a_1, a_2, …, a_n)

begin

i := 1 { i is left endpoint of search interval }

j := n { j is right endpoint of search interval }

while $i < j$

begin

$m := \lfloor (i + j) /2 \rfloor$

if $x > a_m$ **then** $i := m+1$

else $j := m$

end

if $x = a_i$ **then** *location* := i

else *location* := 0

{ *location* is the subscript of the term equal to x, or 0 if x is not found }

end

Algorithm *BubbleSort* ($a_1, \ldots, a_n$)

begin

for $i := 1$ **to** n-1

for $j := 1$ **to** n-i

if $a_j > a_{j+1}$ **then** interchange a_j and a_{j+1}

{ $a_1, \ldots, a_n$ is in increasing order }

end

Algorithm Characteristics

Some important features of algorithms:

- *Input*. Information or data that comes in.
- *Output*. Information or data that goes out.
- *Definiteness*. Precisely defined.
- *Correctness*. Outputs correctly relate to inputs.
- *Finiteness*. Won't take forever to describe or run.
- *Effectiveness*. Individual steps are all do-able.
- *Generality*. Works for many possible inputs.
- *Efficiency*. Takes little time & memory to run.

Informal *statement*

- Sometimes we may write a statement as an informal English imperative, if the meaning is still clear and precise: "swap x and y"
- Keep in mind that real programming languages never allow this.
- When we ask for an algorithm to do so-and-so, writing "Do so-and-so" isn't enough!
  - Break down algorithm into detailed steps.

begin *statements* end

- Groups a sequence of statements together:

```
begin
    statement 1
    statement 2
    …
    statement n
end
```

- Allows sequence to be used like a single statement.
- Might be used:
  1. After a **procedure** declaration.
  2. In an **if** statement after **then** or **else**.
  3. In the body of a **for** or **while** loop.

{*comment*}

- Not executed (does nothing).
- Natural-language text explaining some aspect of the procedure to human readers.
- Also called a *remark* in some real programming languages.
- Example:
  - {Note that v is the largest integer seen so far.}

if *condition* then *statement*

- Evaluate the propositional expression *condition*.
- If the resulting truth value is **true**, then execute the statement *statement*; otherwise, just skip on ahead to the next statement.
- Variant: **if** *cond* **then** *stmt1* **else** *stmt2*
  Like before, but if truth value of *cond* is **false**, then executes *stmt2*.

while *condition* *statement*

- *Evaluate* the propositional expression *condition*.
- If the resulting value is **true**, then execute *statement*.
- Continue repeating the above two actions over and over until finally the *condition* evaluates to **false**; then go on to the next statement.

for <u>*var*</u> := <u>*initial*</u> to <u>*final*</u> <u>*statement*</u>

- <u>*Initial*</u> is an integer expression.
- <u>*Final*</u> is another integer expression.
- Repeatedly execute <u>*statement*</u>, first with variable <u>*var*</u> := <u>*initial*</u>, then with <u>*var*</u> := <u>*initial*</u>+1, then with <u>*var*</u> := <u>*initial*</u>+2, *etc*., then finally with <u>*var*</u> := <u>*final*</u>.
- What happens if <u>*statement*</u> changes the value that <u>*initial*</u> or <u>*final*</u> evaluates to?

for *var* := *initial* to *final* *statement*

- **For** can be exactly defined in terms of **while,** like so:

```
begin
    var := initial
    while var ≤ final
        begin
            statement
            var := var + 1
        end
end
```

Procedure (*argument*)

- A *procedure call* statement invokes the named *procedure*, giving it as its input the value of the *argument* expression.
- Various real programming languages refer to procedures as *functions* (since the procedure call notation works similarly to function application $f(x)$), or as *subroutines*, *subprograms*, or *methods*.

Greedy Algorithms

- Many algorithms are designed to solve optimization problems, and one of the simplest approaches often leads to a solution of an optimization problem
- Algorithms that make what seems to be the *best choice at each step* are called “Greedy Algorithms” instead of considering all sequences of steps.
- But, “Greedy Algorithms” don’t works well for all optimization problems

Exercise

1. Describe an algorithm to find the longest word in an English sentence (where a word is a string of letters and a sentence is a list of words, separated by blanks).

2. Describe an algorithm that locates the first occurrence of the largest element in a finite list of integers, where the integers in the list are not necessarily distinct.

7-2. Complexity of Algorithms

Algorithmic Complexity

- The *algorithmic complexity* of a computation is some measure of how *difficult* it is to perform the computation.
- Measures some aspect of *cost* of computation (in a general sense of cost).
- Common complexity measures:
  1. <u>*"Time" complexity*: *# of ops or steps required*</u>
  2. *"Space" complexity*: *# of memory bits required*

Complexity Depends on Input

- Most algorithms have different complexities for inputs of different sizes. (*E.g.* searching a long list takes more time than searching a short one.)
- Therefore, complexity is usually expressed as a *function* of input length.
- This function usually gives the complexity for the *worst-case* input of any given length.

Orders of Growth

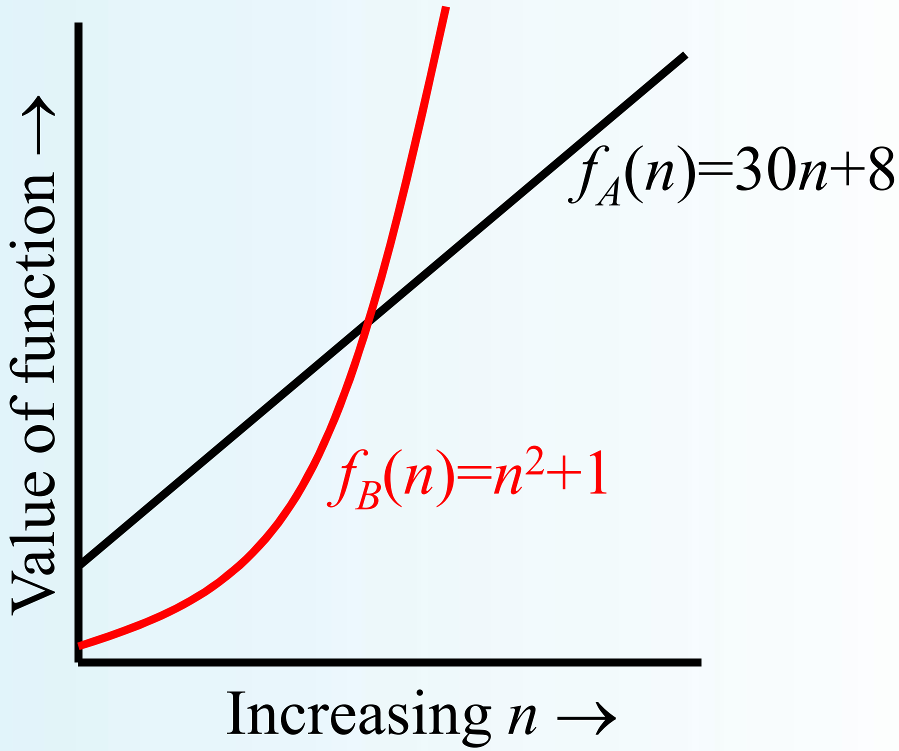
- For functions over numbers, we often need to know a rough measure of how fast a function grows.
- If $f(x)$ is faster growing than $g(x)$, then $f(x)$ always eventually becomes larger than $g(x)$ *in the limit* (for large enough values of x).
- Useful in engineering for showing that one design *scales* better or worse than another.

Orders of Growth - Motivation

- Suppose you are designing a web site to process user data (*e.g.*, financial records).
- Suppose database program ***A*** takes $f_A(n)=30n+8$ microseconds to process any n records, while program ***B*** takes $f_B(n)=n^2+1$ microseconds to process the n records.
- Which program do you choose, knowing you'll want to support millions of users? ***A***.

Visualizing Orders of Growth

- On a graph, as you go to the right, a faster growing function eventually becomes larger...



Concept of Order of Growth

- We say $f_A(n)=30n+8$ is *order n*, or $O(n)$.
  It is, at most, roughly *proportional* to n.
- $f_B(n)=n^2+1$ is *order* n^2, or $O(n^2)$. It is roughly proportional to n^2.
- Any $O(n^2)$ function is faster-growing than any $O(n)$ function.
- For large numbers of user records, the $O(n^2)$ function will always take more time.

O(g), *at most order g*

- *Definition*:

  Let there be a function g:$\boldsymbol{R}\rightarrow\boldsymbol{R}$, The "*at most order g*", written $O(g)$, is defined to be

  $O(g) = \{f{:}\boldsymbol{R}\rightarrow\boldsymbol{R} \mid (\exists c,k)(\forall x>k)(|f(x)| \leq |c \cdot g(x)|)\}$.

  "Beyond some point k, function f is at most a constant c times g (*i.e.,* proportional to g)."

  Note "f is *at most order g*", or "f is $O(g)$", or "f=$O(g)$", all just mean that $f\in O(g)$.

Examples of "Big-O" Proof

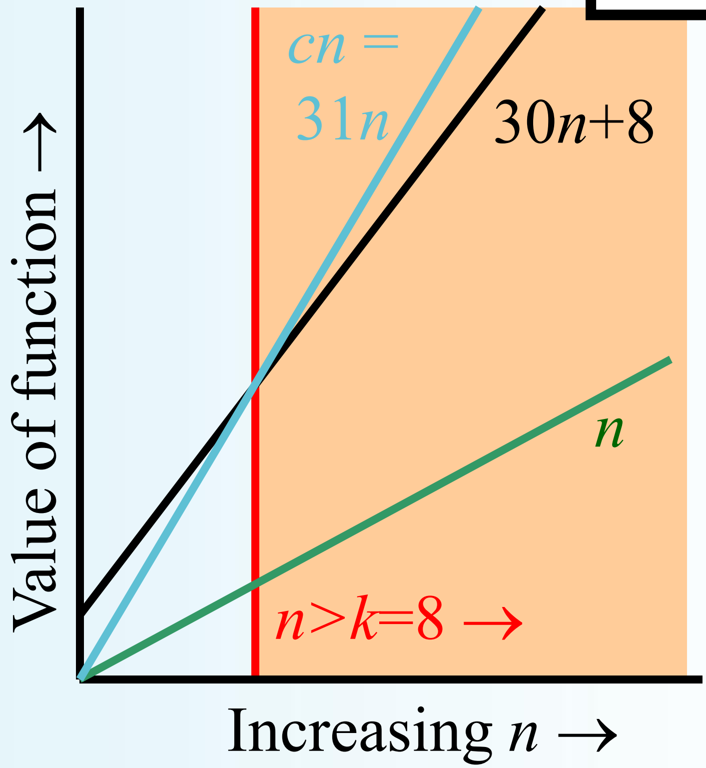
1. Show that $30n+8$ is $O(n)$.
   - Show $(\exists c,k)(\forall n>k)(30n+8 \le cn)$.
     - Let $c=31$, $k=8$. Assume $n>k=8$. Then,
       $cn = 31n = 30n + n > 30n+8$, so $30n+8 < cn$.

2. Show that n^2+1 is $O(n^2)$.
   - Show $(\exists c,k)(\forall n>k)(n^2+1 \le cn^2)$.
     - Let $c=2$, $k=1$. Assume $n>1$. Then,
       $cn^2 = 2n^2 = n^2+n^2 > n^2+1$, or $n^2+1< cn^2$.

Big-O example, graphically

- Note $30n+8$ isn't less than n *anywhere* ($n>0$).
- It isn't even less than $31n$ *everywhere*.
- But it *is* less than $31n$ <u>everywhere to the right of $n=8$</u>.

$30n+8 \in O(n)$



Useful Facts about Big-O

1. Big-O, as a relation, is transitive:
   $f \in O(g) \wedge g \in O(h) \rightarrow f \in O(h)$
2. O with constant multiples, roots, and logs...
   $\forall f$ (in $\Omega(1)$) & constants $a,b \in \boldsymbol{R}$, with $b \geq 0$,
   af, f^{1-b}, and $(\log_b f)^a$ are all $O(f)$.
3. Sums of functions:
   If $g \in O(f)$ and $h \in O(f)$, then $g+h \in O(f)$.

4. $\forall c>0,\ O(cf)=O(f+c)=O(f-\mathrm{c})=O(f)$

5. $f_1 \in O(g_1) \wedge f_2 \in O(g_2) \rightarrow$

   - $f_1 \cdot f_2 \in O(g_1 g_2)$
   - $f_1+f_2 \in O(g_1+g_2)$

     $= O(\max(g_1, g_2))$

     $= O(g_1)$ if $g_2 \in \mathrm{O}(g_1)$

Order of Growth Expressions

- "$O(f)$" when used as a term in an arithmetic expression means: "some function f such that $f \in O(f)$".
- *E.g.*: "$x^2+O(x)$" means "x^2 plus some function that is $O(x)$".
- Formally, you can think of any such expression as denoting a set of functions:

  "$x^2+O(x)$" $= \{g \mid (\exists f \in O(x))(g(x)= x^2+f(x))\}$

Order of Growth Equations

- Suppose E_1 and E_2 are order-of-growth expressions corresponding to the sets of functions S and T, respectively.
- Then the "equation" $E_1 = E_2$ really means

  $$(\forall f \in S)(\exists g \in T)(f = g)$$

  or simply $S \subseteq T$.
- Example: $x^2 + O(x) = O(x^2)$ means

  $$(\forall f \in O(x))(\exists g \in O(x^2))(x^2 + f(x) = g(x))$$

Useful Facts about Big-O

- $\forall\ f,g$ & constants $a,b\in\mathbf{R}$, with $b\geq 0$,
  1. $af = O(f)$; (*e.g.* $3x^2 = O(x^2)$)
  2. $f+O(f) = O(f)$; (*e.g.* $x^2+x = O(x^2)$)
- Also, if $f=\Omega(1)$ (*at least* order 1), then:
  1. $|f|^{1-b} = O(f)$; (*e.g.* $x^{-1} = O(x)$)
  2. $(\log_b |f|)^a = O(f)$. (*e.g.* $\log x = O(x)$)
  3. $g=O(fg)$ (*e.g.* $x = O(x \log x)$)
  4. $fg \neq O(g)$ (*e.g.* $x \log x \neq O(x)$)
  5. $a=O(f)$ (*e.g.* $3 = O(x)$)

$\Omega(g)$, *at least order g*

- *Definition*:

  Let there be *a* function g: $\boldsymbol{R}\rightarrow\boldsymbol{R}$. The "*at least order g*", written $\Omega(g)$, is defined to be:

  $\Omega(g) = \{f{:}\boldsymbol{R}\rightarrow\boldsymbol{R} \;\boldsymbol{|}(\exists c,k)(\forall x>k)(|f(x)| \geq |cg(x)|)\}$.

  "Beyond some point k, function f is at least a constant c times g (*i.e.,* proportional to g)."

  Note "f is *at least order g*", or "f is $\Omega(g)$", or "$f = \Omega(g)$", all just mean that $f\in\Omega(g)$.

$\Theta(g)$, *exactly order g*

- *Definition*:

  Let there be *a* function g: $\boldsymbol{R}\rightarrow\boldsymbol{R}$. The "*exactly order g*", written $\Theta(g)$, is defined to be:

  $\Theta(g) = \{f{:}\boldsymbol{R}\rightarrow\boldsymbol{R} \mid (\exists c_1 c_2 k)(\forall x>k)(|c_1 g(x)|\leq|f(x)|\leq|c_2 g(x)|)\}$.

  "Everywhere beyond some point k, $f(x)$ lies in between two multiples of $g(x)$."

  Note *"g and f are of the same order*", or *"f* is $\Theta(g)$*",* or "*f is (exactly) order g*", all just mean that $f\in\Theta(g)$.

Rules for Θ

- Mostly like rules for $O(\ )$, except:
- $\forall\ f,g>0$ & constants $a,b\in \boldsymbol{R}$, with $b>0$,
  $af \in \Theta(f)$, but ← Same as with O.
  $f \notin \Theta(fg)$ unless $g=\Theta(1)$ ← Unlike O.
  $|f|^{1-b} \notin \Theta(f)$, and ← Unlike with O.
  $(\log_b |f|)^c \notin \Theta(f)$. ← Unlike with O.
- The functions in the latter two cases we say are *strictly of lower order* than $\Theta(f)$.

Example of Θ

- Determine whether: $\left(\sum_{i=1}^{n} i\right) \overset{?}{\in} \Theta(n^2)$

- Quick solution:

$$\left(\sum_{i=1}^{n} i\right) = n(n-1)/2$$
$$= n \cdot \Theta(n)/2$$
$$= n \cdot \Theta(n)$$
$$= \Theta(n^2)$$

Complexity Analysis

Now, what is the simplest form of the exact (Θ) order of growth of $t(n)$?

$$t(n) = t_1 + \left(\sum_{i=2}^{n} (t_2 + t_3) \right) + t_4$$

$$= \Theta(1) + \left(\sum_{i=2}^{n} \Theta(1) \right) + \Theta(1) = \Theta(1) + (n-1)\Theta(1)$$

$$= \Theta(1) + \Theta(n)\Theta(1) = \Theta(1) + \Theta(n) = \Theta(n)$$

Names for some orders of growth

- $\Theta(1)$ Constant
- $\Theta(\log_c n)$ Logarithmic (same order $\forall c$)
- $\Theta(\log^c n)$ Polylogarithmic
- $\Theta(n)$ Linear
- $\Theta(n^c)$ Polynomial
- $\Theta(c^n)$, $c>1$ Exponential
- $\Theta(n!)$ Factorial

(With c a constant.)

Problem Complexity

- The complexity of a computational *problem* or *task* is (the order of growth of) the complexity of <u>the algorithm with the lowest order of growth of complexity</u> for solving that problem or performing that task.
- *E.g.* the problem of searching an ordered list has *at most logarithmic* time complexity. (Complexity is $O(\log n)$.)

Tractable *vs.* Intractable

- A problem or algorithm with at most polynomial time complexity is considered *tractable* (or *feasible*). **P** is the set of all tractable problems.
- A problem or algorithm that has more than polynomial complexity is considered *intractable* (or *infeasible*).
- Note that $n^{1,000,000}$ is *technically* tractable, but really impossible. $n^{\log\log\log n}$ is *technically* intractable, but easy. Such cases are rare though.

Unsolvable problems

- Turing discovered in the 1930's that there are problems unsolvable by *any* algorithm.
  - Or equivalently, there are undecidable yes/no questions, and uncomputable functions.
- Example: the *halting problem*.
  - Given an arbitrary algorithm and its input, will that algorithm eventually halt, or will it continue forever in an "*infinite loop*?"

P *vs.* NP

- **NP** is the set of problems for which there exists a tractable algorithm for *checking solutions* to see if they are correct.

  ex : The satisfiability problem of a compound proposition

- We know **P**$\subseteq$**NP**, but the most famous unproven conjecture in computer science is that this inclusion is *proper* (*i.e.*, that **P**$\subset$**NP** rather than **P**=**NP**).

Computer Time Examples

	(1.25 bytes)	(125 kB)
$\#ops(n)$	n=10	n=10^6
$\log_2 n$	3.3 ns	19.9 ns
n	10 ns	1 ms
$n \log_2 n$	33 ns	19.9 ms
n^2	100 ns	16 m 40 s
2^n	1.024 μs	$10^{301,004.5}$ Gyr
$n!$	3.63 ms	Ouch!

Assume time = 1 ns (10^{-9} second) per op, problem size = n bits, #ops a function of n as shown.

Exercise

1. Prove the following:

   (a) $n \cdot \sin n$ is $O(\text{n})$.

   (b) $x \cdot \log x$ is $O(x^2)$ but that x^2 is not $O(x \cdot \log x)$.

   (c) The function $f(n)=2n^2-n-1$ is $O(n^2)$.

2. Write the algorithm that puts the first four terms of a list of arbitrary length in increasing order, and show that this algorithm has time complexity $O(1)$ in terms of the number of comparisons used.

Discrete Mathematics

8. Probability & Random Variables

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

8-1. Probability

Why Probability?

- In the real world, we often don’t know whether a given proposition is true or false.
- Probability theory gives us a way to reason about propositions whose truth is *uncertain*.
- Useful in weighing evidence, diagnosing problems, and analyzing situations whose exact details are unknown.

Definitions

- Sample point:

  A representation of a possible outcome of an experiment
- Sample space:

  The totality of all possible samples points, that is, the representation of all possible outcomes of an experiment
- Event:

  A collection of outcomes or a set of sample points

Events

- *Definition*:

  An *event* E is a set of possible outcomes:

  $$E \subseteq S$$

  where S is the sample space.

Probability

- *Definition*:

  The *probability,* $\Pr[E] \in [0,1]$, of an event E is a real number representing our degree of certainty that E will occur.

  1. If $\Pr[E] = 1$, then E is absolutely certain to occur.
  2. If $\Pr[E] = 0$, then E is absolutely certain *not* to occur.
  3. If $\Pr[E] = ½$, then we are *completely uncertain* about whether E will occur.

Probability Distribution

- *Definition*:

  Let p be any function, $p{:}S\rightarrow[0,1]$, such that

  1. $0 \le p(w) \le 1$ for every outcome, $w\in S$.
  2. $\sum_{w\in S} p(w) = 1.$

  Such a p is called a *probability distribution*.

  Then the probability of any event $E\subseteq S$ is

  $$\Pr[E] = \sum_{w\in E} p(w)$$

Probability of Complementary Events

- *Theorem*:

  Let E be an event in a sample space S. Then, the probability of the *complementary* event $\overline{E}$ is

  $\Pr[\overline{E}] = 1 - \Pr[E]$

Probability of Unions of Events

- *Theorem*:

  Let $E_1, E_2 \subseteq S$. Then
  $\Pr[E_1 \cup E_2] = \Pr[E_1] + \Pr[E_2] - \Pr[E_1 \cap E_2]$

  Proof:

  By the inclusion-exclusion principle.

Mutually Exclusive Events

- *Definition*:

  Two events E_1, E_2 are called *mutually exclusive* if they are disjoint: $E_1 \cap E_2 = \varnothing$

- *Theorem*:

  For mutually exclusive events, E_1 and E_2,
  $\Pr[E_1 \cup E_2] = \Pr[E_1] + \Pr[E_2]$.

Exhaustive Sets of Events

- *Definition*:
  1. A set $\boldsymbol{E} = \{E_1, E_2, \ldots\}$ of events in the sample space S is *exhaustive* if $\bigcup E_i = S$
  2. An exhaustive set of events that are all mutually exclusive with each other has the property that

$$\sum \Pr[E_i] = 1$$

Independent Events

- *Definition*:

  Two events E,F are *independent* if

  $\Pr[E \cap F] = \Pr[E] \cdot \Pr[F]$.

- Example: Flip a coin, and roll a die. Then,

  Pr[quarter is head $\wedge$ die is 1] =

  Pr[quarter is head] $\times$ Pr[die is 1].

Conditional Probability

- *Definition*:

  Let E, F be events such that $\Pr[F]>0$. Then, the *conditional probability of E given F*, written $\Pr[E|F]$, is defined to be $\Pr[E|F] = \Pr[E\cap F]/\Pr[F]$.

- *Theorem*:

  If E and F are independent, $\Pr[E|F] = \Pr[E]$.

Bayes's Theorem

- *Theorem*:

  The probability that a hypothesis H is correct, given data D, is

$$\Pr[H \mid D] = \frac{\Pr[D \mid H] \cdot \Pr[H]}{\Pr[D]}$$

Proof:

From the definition of conditional probability.

8-2. Random Variables

Random Variables

Let X be a single-valued real function, $X{:}S \rightarrow T$, where S is a sample space and T is a set of real numbers.
Consider the range of X, denoted by R_X, to be a new sample space, S_X. The probability of the event A in the new sample space is then given by $\Pr[A \subseteq S_X] \equiv \Pr[X^{-1}(A) \subseteq S] \equiv \Pr[X{=}A]$.

Whenever a function X defined on a sample space S is such that the probability of the inverse image $X^{-1}(A)$ is defined for each event A in the range sample space S_X,
Then the function X is said to be a *measurable function* on S and is called a *random variable*.
(Note a random variable is in fact a function and not a variable.)

Random Variables

1. If the range is a continuum, it is called a *continuous random variable.*
2. If the range consists only of isolated points, it is called a *discrete random variable*.
3. If the range is a combination of both continuum parts and isolated points, it is called a *mixed random variable.*

Experiments

- *Definition*:
  1. A (stochastic) *experiment* is a process by which a given random variable gets a specific value.
  2. The *sample space S* of the experiment is the domain of the random variable.
  3. The *outcome* of the experiment is the specific value of the random variable that is selected.

Expected Values

- *Definition*:

  The *mean*, or the *expectation*, or *expected value* of the discrete random variable X is given by

  $$E[X] = \sum_{x_k \in range(X)} x_k \cdot \Pr[X=x_k].$$

- *Theorem*:

  Let X_1, X_2 be any two random variables derived from the same sample space. Then,

  1. $E[X_1+X_2] = E[X_1] + E[X_2]$
  2. $E[aX_1 + b] = aE[X_1] + b$

Independent Random Variables

- *Definition*:

  Two random variables X and Y are independent if

  $\Pr(X=r_1 \text{ and } Y=r_2) = \Pr(X=r_1) \cdot \Pr(Y=r_2)$ for every real numbers, r_1 and r_2

- *Theorem*:

  If X and Y are independent random variables, then

  $E(XY) = E(X) \cdot E(Y)$

Variance

- *Definition*:
  1. The *variance* $Var[X] = \sigma^2(X)$ of a random variable X is the expected value of the square of the difference between the value of X and its expected value $E[X]$:

$$Var[X] = E[(X - E[X])^2]$$

  2. The *standard deviation* of X, $\sigma(X) = Var[X]^{1/2}$.

Probability Distribution of a Random Variable

- *Definition*:

1. The distribution of a discrete random variable, X, is a set of pairs, $(r, \Pr[X=r])$, for each r in range(X).

2. The distribution of a continuous random variable, X, is given by a density function, $f_X(x)$, where

$$\Pr[X \in (a,b]] = \int_a^b f_X(u)du$$

Binomial Distribution

- The probability, $P(k)$, of exactly k successes in n independent Bernoulli trials, with probability of success p and probability of failure q=1-p, is

$$\frac{n!}{k!(n-k)!}p^k q^{n-k}$$

- If a random variable X follows a Binomial distribution, then Pr[X=k] = $P(k)$ where range(X) = {0, 1, 2,…, n}.

- *Theorem*:

  Let X be a random variable with a binomial distribution.

  Then

$$E[X]=np \quad \text{and} \quad Var[X]=np(1-p)$$

Gaussian (Normal) Distribution

- A Gaussian distribution is a bell-shaped distribution defined by the probability density function

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

- If a random variable X follows a Gaussian distribution, then

$$E(X) = \mu \quad \text{and} \quad Var(X) = \sigma^2$$

Central Limit Theorem

Let $X_1, \ldots, X_n$ be *n independent random variables obeying the same unknown probability distribution with mean* μ *and finite variance* σ^2. Then the probability distribution of the sample mean,

$$Y_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

approaches a Gaussian distribution as $n \to \infty$, where the mean of Y_n approaches μ and the standard variance approaches $\frac{\sigma^2}{n}$.

Exercise

1. Let A, B and C be events in a sample space and suppose $\Pr(A \cap B) \neq 0$. Prove that $\Pr(A \cap B \cap C) = \Pr(A) \cdot \Pr(B|A) \cdot \Pr(C|\, A \cap B)$

2. Let A and B be events with nonzero probability in a sample space.

   (a) Suppose $\Pr(A|B) > \Pr(A)$. Must it be the case that $\Pr(B|A) > \Pr(B)$?

   (b) Suppose $\Pr(A|B) < \Pr(A)$. Must it be the case that $\Pr(B|A) < \Pr(B)$?

3. Let X and Y be two independent random variables.

(a) Give the definition of variance, $Var(X)$, of X and show that $Var(X)= E(X^2)-E(X)^2$.

(b) Show that $Var(X+Y)=Var(X)+Var(Y)$.