1 Sets and Logic

1.1 Sets

*An **unordered**(distinguished) **collection** of **objects***

*An **object** in a set is called an **element** or a **member** of the set.*

*(An **ordered**(undistinguished) **collection** of **objects***

orderd pair, triple, quaduruple, …, n-tuple)

Two ways to define sets

*i) To **enumerate** the elements(원소나열법)* **(1.1.1)**

$A = \{a_1, a_2, \ldots, a_n\}$ ***finite** sets*

$A = \{a_1, a_2, \ldots\}$ ***infinite** sets*

*ii) To specify **conditions**(조건제시법)* **(1.1.2)**

$A = \{x | p(x)\}$ *p(x): **predicate**(see **Sec. 1.5 Quantifiers**)*

$A = \{x \in U | p(x)\}$ *U: **universe**(domain) of discourse*

iii) to write a program(?; CS322)

Some important sets in discrete mathematics(p 2)

$\mathbf{Z} = \{\ldots, -2, -1, 0, 1, 2, 3, \ldots\}$ — *set of* ***integers***.

$\mathbf{Q} = \{p/q \mid p \in \mathbf{Z}, q \in \mathbf{Z} \text{ and } q \neq 0\}$ — *set of* ***rational numbers***.

$\mathbf{R}$ — *set of* ***real numbers***.

$\mathbf{Z}^+ = \{1, 2, 3, \ldots\}$ — *set of* ***positive natural numbers***.

$\mathbf{N} = \{0, 1, 2, 3, \ldots\}$ — *set of* ***natural numbers***.

$\mathbf{N}_n = \{n, n+1, \ldots\}$ $\quad n \in \mathbf{Z}$

Def. Cardinality of a set(p 3)

$|A|$ *the* ***number*** *of elements in the set A*

A set ***contain*** *its* ***elements****.(p 3)*

$x \in X$ "*an* ***object*** *x is an* ***element*** *of the* ***set*** *X*"

$x \notin X$ "*an* ***object*** *x is* ***not*** *an* ***element*** *of the* ***set*** *X*"

Def. Empty set(p 3)

The set with no elements is ***empty****(****null*** *or* ***void****) set denoted* $\varnothing$ *or* $\{\}$.

Note that $|\varnothing| = |\{\}| = 0$.

Def. Equality of two sets(p 3)

Two sets X and Y are ***equal*** *and we write* $X = Y$,

If X and Y have the ***same*** *elements.*

Otherwise

a set X is ***not*** *equal to Y(written* $X \neq Y$*).*

Note that $\{\} = \varnothing \neq \{\varnothing\}$.

Def. Subsets(p 4)

The set X is said to be a ***subset*** *of Y(written* $X \subseteq Y$*),*

if every elements of X is also an elements of Y.

Or

*If for **every** x, if* $x \in X$, *then* $y \in Y$.

Def. Equality of two sets, revisited

$X = Y$ if $X \subseteq Y \wedge Y \subseteq X$.

i) For **every** x, if $x \in X$, then $x \in Y$, and — $X \subseteq Y \wedge$

ii) For **every** x, if $x \in Y$, then $x \in X$. — $Y \subseteq X$

Note that for **every** set X, $X \subseteq X$ and $\emptyset \subseteq X$.

Def. Proper subsets(p 5; 진부분집합)

If $X \subseteq Y$ and(but) $Y \not\subseteq X$, then we write $X \subset Y$ and

we say that X is a **proper subset** of Y, and we write $X \subset Y$.

Note that for **every** set X, $X \not\subset X$.

Def. Power Set of a Set(p 5)

*The **set**(**collection**) of **all subsets** of a set A, denoted P(A) or 2^A, is the **power set** of A.* $P(A) = 2^A = \{B \mid B \subseteq A\}$

***Note** a **collection**(set) may have a set as an **elemement** of the **collection**.*

$$X \subseteq A \quad \Leftrightarrow \quad X \in 2^A.$$

***Exa. 1.1.13** Let A = {a, b, c}. Then*

$$P(\{a, b, c\}) = 2^{\{a, b, c\}} = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$$

$$|P(\{a, b, c\})| = |2^{\{a, b, c\}}| = 8.$$

$0_c0_b0_a \leftrightarrow \emptyset$ $\quad 0_c0_b1_a \leftrightarrow \{a\}$ $\quad 0_c1_b0_a \leftrightarrow \{b\}$ $\quad 0_c1_b1_a \leftrightarrow \{a, b\}$

$1_c0_b0_a \leftrightarrow \{c\}$ $\quad 1_c0_b1_a \leftrightarrow \{a, c\}$ $\quad 1_c1_b0_a \leftrightarrow \{b, c\}$ $\quad 1_c1_b1_a \leftrightarrow \{a, b, c\}$

***isomorphic**(See **Sec. 3.1**) to the binary number of length 3*

$$|\{a, b, c\}| = 3 \qquad 8 = 2^3.$$

$|2^A| = |P(A)| \overset{?}{=} 2^{|A|}$ *See **Sec 2.4 Mathematical Induction(Thm. 2.4.6)***

<u>Definitions of set operations</u> *Let X and Y be two sets. Then(p 6)*

1. ***<u>Union</u>*** *of X and Y* $X \cup Y = \{x \mid x \in X \vee x \in Y\}$
2. ***<u>Intersection</u>*** *of X and Y* $X \cap Y = \{x \mid x \in X \wedge x \in Y\}$
3. ***<u>Difference</u>*** *of X and Y* $X - Y = \{x \mid x \in X \wedge x \notin Y\}$

   Relative complement *of Y from X*
4. *Let U be a* ***universe(universal set****) and* $X \subseteq U$. *Then(p 7)*

   <u>Absolout complement</u> *of X* $\overline{X} = U - X = \{x \in U \mid x \notin X\}$

Two sets X and Y are ***<u>disjoint</u>****,*

if their ***intersection*** *is* ***empty****, i.e.* $A \cap B = \varnothing$.

Let S be a ***collection*** *of sets. The* ***collection*** *S is* ***<u>pairwise disjoint</u>****(p 6)*

Every pair of set X and Y in S are ***disjoint****, i.e.* $X \cap Y = \varnothing$.

<u>Venn diagram</u>*(p 7)*

pictorial *view of sets with* ***universe***

n ***sets*** 2^n ***regions***

Thm. 1.1.21 Algebraic Rules of Set Equalites(*p 8-9*)

Let U be universal set and A, B, and C be ***any*** *subsets of U.*

$(\forall A, B, C \subseteq U \equiv \forall A, B, C \in 2^U)$

1. **Associative laws** $(A \cup B) \cup C = A \cup (B \cup C)$ $(A \cap B) \cap C = A \cap (B \cap C)$
2. **Commutative laws** $A \cup B = B \cup A$ $A \cap B = B \cap A$
3. **Distributive...** $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
4. **Identity laws** $A \cup \varnothing = A$ $A \cap U = A$
5. **Complement laws** $A \cup \overline{A} = U$ $A \cap \overline{A} = \varnothing$
6. **Idempotent laws** $A \cup A = A$ $A \cap A = A$
7. **Domination(Bound) laws** $A \cup U = U$ $A \cap \varnothing = \varnothing$
8. **Absorption laws** $A \cup (A \cap B) = A$ $A \cap (A \cup B) = A$
9. **Involution law** $\neg\overline{(A)} = A$
10. **0/1 laws** $\overline{\varnothing} = U$ $\overline{U} = \varnothing$
11. **De Morgan's laws** $\overline{A \cup B} = \overline{A} \cap \overline{B}$ $\overline{A \cap B} = \overline{A} \cup \overline{B}$

Proof ***insight for proof*** *using* ***Venn diagram***

Let S be a ***collection*** *of sets. Then we can write*

$\cup S = \{a \mid \text{for } \textbf{some } A \in S \text{ .∍. } a \in A\}$ $\cap S = \{a \mid \text{for } \textbf{all } A \in S, a \in A\}$

$\cup S = \{a \mid \exists A \in S \text{ .∍. } a \in A\} \neq S$ $\cap S = \{a \mid \forall A \in S, a \in A\}$

Let $S = \{A_1, A_2, \ldots, A_n\}$ *be a* ***collection*** *of n-sets. Then we can write*

$\cup_{i=1}^{n} A_i = A_1 \cup A_2 \cup \ldots \cup A_n$, $\cap_{i=1}^{n} A_i = A_1 \cap A_2 \cap \ldots \cap A_n$.

i: ***index variable***

Let $I = \{1, 2, \ldots, n\}$. *Then* *I:* ***index set***

$\cup_{i \in I} A_i = \cup_{i=1}^{n} A_i$, $\cap_{i \in I} A_i = \cap_{i=1}^{n} A_i$.

Unions *and* ***Intersections*** *are* ***assosiative(semigroup)***

If an operation is ***associative****,*

the ***binary*** *operation* ***extends*** *to* ***n-ary*** *operation*

Let $x_1, x_2, \ldots, x_n \in \mathbf{R}$. *Then*

$$\sum_{i=1}^{n} x_i = x_1 + x_2 + \ldots + x_n, \qquad \prod_{i=1}^{n} x_i = x_1 \cdot x_2 \cdot \ldots \cdot x_n.$$

Compare $\sum_{i=1}^{n} x_i$ *with* $\sum_{i \in I} x_i$ *where* $I = \{1, 2, \ldots, n\}$.

for *i:=1* ***to*** *n* ***do*** Σx_i ***od*** ***for*** $i \in I$ ***do*** Σx_i ***od***

Def. *A* ***partion*** *of a set A is an* ***exhaustive pairwise disjoint collection*** *of* ***subsets*** *of A.*

$$Par(A) = \{A_1, A_2, \ldots, A_n\} \text{ such that}(.\ni.)$$

i) $\cup_{i \in N_n} A_i = A$ ***exhaustive*** $(1 \leq \forall i \leq n\text{: } A_i \subseteq A)$

ii) $\cap_{i \neq j \in N_n} (A_i \cap A_j) = \varnothing \Sigma)$ ***pairwise disjoint***

Note that $\cup_{A \in Par(A)} Par(A) = A \wedge \cap_{A \in Par(A)} Par(A) = \varnothing$. *BUT*

$\cup_S S = S \wedge \cap_S S = \varnothing$ *does not* ***gurantee*** *the collection S to be a partition.*

Cartesian product of two sets *(p10)*

Def. *An* ***ordered pair*** *(순서쌍) of elements.*

$(a, b) \neq (b, a)$ but $\{a, b\} = \{b, a\}$.

Def. *Let A and B be two sets. We define* ***Cartesian product*** *of A and B as*

$A \times B = \{(a, b) | a \in A, b \in B\}$

(a, b) an ***ordered pair***

$^{?}X \times Y$ *X-Y* 평면 (중)

(Example) Let $A = \{a, b\}$ *and* $B = \{1, 2, 3\}$. *Then*

$A \times B = \{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$

$|A \times B| = 2 \times 3 = |A| \times |B|$

$(A \times B) \times C \neq A \times (B \times C)$, *since* $((a, 1), \alpha) \neq (a, (1, \alpha))$.

BUT

Def. *The* ***Cartesian product*** *of the n sets* $A_1, A_2, \ldots, A_n$, *denoted by*

$$A_1 \times A_2 \times \ldots \times A_n = \{(a_1, a_2, \ldots, a_n) | a_i \in A_i \text{ for } i = 1, 2, \ldots, n\}$$

Def. *The* ***ordered n-tuple*** $(a_1, a_2, \ldots, a_n)$ *is the* ***ordered collection****, that has* a_1 *as its first element,* a_2 *as its second element, …,* a_n *as its n-th element.*

$$(a_1, a_2, \ldots, a_n) = (b_1, b_2, \ldots, b_n), \text{ iff, } a_i = b_i \text{ for } i = 1, 2, \ldots, n.$$

$(a_1, a_2, \ldots, a_n) = (b_1, b_2, \ldots, b_n) \equiv (a_1 = b_1) \wedge (a_2 = b_2) \wedge \ldots \wedge (a_n = b_n)$.

Note that $(A \times B) \times C \neq A \times (B \times C) \neq A \times B \times C$.

BUT $|(A \times B) \times C| = |A \times (B \times C)| \leftrightarrow |A \times B \times C|$

BUT $(A \times B) \times C \leftrightarrow A \times (B \times C) \leftrightarrow A \times B \times C$ *See* ***Sec. 3.1***

***Four cases*(binary *relations*) between two sets A and B.**

1. ***equal*** $A = B$
2. ***subset*** $A \subseteq B$
   2.1. ***proper subset*** $A \subset B$
3. ***disjoint*** $A \cap B = \varnothing$
4. ***incomparable*** *Otherwise*

Four regions *in the* ***Venn diagram*** *of* ***two*** *sets A and B.*

i) $A \cap B$ ii) $\overline{A} \cap B$ iii) $A \cap \overline{B}$ iv) $\overline{A} \cap \overline{B}$

Truth table *in* ***predicate logic*** *with* ***two*** *variables*

Four cases *for between two set and for* ***regions*** *in theVenn diagram*

i) $(A \cap \overline{B} = \varnothing) \wedge (\overline{A} \cap B = \varnothing) \equiv A = B$, ***equal***

ii) $(A \cap \overline{B} = \varnothing)$ or $(\overline{A} \cap B = \varnothing) \equiv A \subseteq B$ or $B \subseteq A$, ***subset***

iii) $(A \cap B = \varnothing) \equiv A \cap B = \varnothing$, ***disjoint***

iv) *otherwise* ***incomparable***

1.2 Propositions

***Propositions*(명제)**

a declarative ***sentence*** *that is either* ***true*** *or* ***false****,*
but ***not both nor neither***

letters denoting propositions $p, q, r, s, \ldots$

T*: true value*

F*: false value*

propositional calculus *or* ***propositional logic***

compounded propositions

propositions *that are* ***formed*** *from* ***existing*** *propositions*
using ***logical operators*(connectives)**

Def. 1.2.1; 3 Conjunction(And)

*Let p and q be a propositions, $p \wedge q$ is a (compounded) proposition, called **conjunction** of p and q, or "p **and** q".*

truth table

Def. 1.2.1; 6 Disjunction(or; Inclusive or)

*Let p and q be a propositions, $p \vee q$ is a proposition, called **disjunction** of p and q, or "p or q".*

Def. 1.2.9 Negation(not)

*Let p be a proposition, $\neg p$ is a (new) **compounded** proposition, called **negation** of p, or "**not** p".*

***Compare** the **truth tables** for $\wedge$, $\vee$, and $\neg$ with the **Venn diagrams** for $\cap$, $\cup$, and $\overline{}$ in **set operations**(**Sec. 1.1**).*

1.2.13' Algebraic Rules of Logical Equivalences

1. ***Associative laws*** $(p \vee q) \vee r = p \vee (q \vee r)$ $(p \wedge q) \wedge r = p \wedge (q \wedge r)$
2. ***Commutative laws*** $p \vee q = q \vee p$ $p \wedge q = q \wedge p$
3. ***Distributive laws*** $p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) = (p \wedge q) \vee (p \wedge r)$
4. ***Identity laws*** $p \vee \mathbf{F} = p$ $p \wedge \mathbf{T} = p$
5. ***Complement laws*** $p \vee \neg p = \mathbf{T}$ $p \wedge \neg p = \mathbf{F}$
6. ***Idempotent laws*** $p \vee p = p$ $p \wedge p = p$
7. ***Domination laws*** $p \vee \mathbf{T} = \mathbf{T}$ $p \wedge \mathbf{F} = \mathbf{F}$
8. ***Absorption laws*** $p \vee (p \wedge q) = p$ $p \wedge (p \vee q) = p$
9. ***Double negation law*** $\neg(\neg p) = p$
10. **T/F *laws*** $\neg \mathbf{T} = \mathbf{F}$ $\neg \mathbf{F} = \mathbf{T}$
11. ***De Morgan's laws*** $\neg(p \vee q) = \neg p \wedge \neg q$ $\neg(p \wedge q) = \neg p \vee \neg q$

Compare with the ***algebraic rules of set equalities*** *in p. 6(**Thm 1.1.21**)*

$(\{\mathbf{T}, \mathbf{F}\}, \vee, \wedge, \neg, \equiv) \leftrightarrow (\{U, \varnothing\}, \cup, \cap, \overline{\ }, =)$

isomorphism *betwween positional logic and set algebra*

Syntax *(문법) of* ***composite*** *propositions*

$\neg$	***negation***	***unary prefix*** *operator*
$\wedge$, $\vee$	***conjunctive, disjunctive***	***binary infix*** *operator*
(,)		***precedence*** *of operator*
T, F		***constant***
p, q, r, s		***variables***

Syntactic ***grammars*** *for propositional logic, set operation and 식*

p ::= **T** | **F** | v | $p \vee p$ | $p \wedge p$ | $\neg p$ | (p)

S ::= $\emptyset$ | U | S | $S \cup S$ | $S \cap S$ | $\overline{S}$ | (S)

식 ::= 수 | v | 식 + 식 | 식 * 식 | – 식 | (식)
| *sin*(식) | 식[식] | *log*(식)

1.3 Conditional Propositions and Logical Equivalence

Def. 1.3.1 *Let p and q be prpositions, the* ***proposition***

"if p then q" (1.3.2) *"p, only if p"*

conditional proposition*(or* ***implication****) written* $p \to q$,

p: ***hypothesis****(or* ***antecedent****;* ***premise****; 가정) of* $p \to q$.

*q:****conclusion****(or* ***consequence****; 결론) of* $p \to q$.

Def. 1.3.3 Truth table *for* ***implication*** $p \to q$.

p	q	$p \to q$
T	**T**	**T**
T	**F**	**F**
F	**T**	**T**
F	**F**	**T**

If p = ***F****, then* $p \to q$ *=* ***T*** *regardless of whether q is* ***T*** *or* ***F****.*

true by default *or* ***vacuously true****.*

An ***implication*** $p \to q$ *says* ***nothing*** *when the* ***hypothesis*** *p is* ***false***

Definition 1.3.8 biconditional(equivalence)

Let p and q be a propositions, $p \leftrightarrow q$ is a proposition, called **biconditional** *of p and q, or "p, if and only if, q".*

Def. 1.3.8' Truth table *for* ***implication*** $p \leftrightarrow q$.

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

if
  $p = q \rightarrow p \leftrightarrow q = \mathbf{T}$
$|\ p \neq q \rightarrow p \leftrightarrow q = \mathbf{F}$
fi

Def. 1.3.10 *Let two propositions P and Q are made up of the n propositions $p_1, p_2, \ldots, p_n$. We say P and Q are* **logically equivalent** *and write $P \equiv Q$, if given any truth value of $p_1, p_2, \ldots, p_n$, either P and Q are both* ***true*** *or P and Q are both* ***false****.*

Def. 1.3.16 *The* ***contrapositive****(****transposision****; 대 우) of the* ***conditional proposition*** $p \to q$ *is* $\neg q \to \neg p$.

$q \to p$(***converse***; 역)
$\neg p \to \neg q$(***inverse***; 이)

Thm. 1.3.18 *The* ***conditional proposition*** $p \to q$ *is and its* ***contrapositive*** $\neg q \to \neg p$ *are* ***logically equivalent.***

proof *The* ***truth table***

p	q	$p \to q$	$\neg q \to \neg p$
T	***T***	***T***	***T***
T	***F***	***F***	***F***
F	***T***	***T***	***T***
F	***F***	***T***	***T***

Def. *A (compound) proposition that is always true:* ***tautology***
A proposition that is always false: ***contradiction***
Otherwise: ***contingency***.

If an ***implication*** $p \rightarrow q$ *is* ***always true(tautology)***, *we write* $p \Rightarrow q$.

$p \Rightarrow q$ — p *is a* ***sufficient condition*** *to be* q.

$\neg q \Rightarrow \neg p$ — q *is a* ***necessary condition*** *to be* p.

If a ***biconditional*** $p \leftrightarrow q$ *is* ***a tautology***, *we write* $p \Leftrightarrow q$ (or $p \equiv q$)

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

$p \rightarrow q \equiv \neg p \vee q$ — *disjunctive normal form*

$p \leftrightarrow q \equiv (\neg p \vee q) \wedge (\neg q \vee p)$ — *disjunctive normal form*

$\equiv (\neg p \wedge \neg q) \vee (\neg p \wedge p) \vee (q \wedge \neg q) \vee (q \wedge p)$ *distribution law*

$\equiv (\neg p \wedge \neg q) \vee \mathbf{F} \vee \mathbf{F} \vee (q \wedge p) \equiv (\neg p \wedge \neg q) \vee (p \wedge q)$

$\equiv (p \equiv q) \equiv (p = q)$. *We will use* $=$ *instead of* $\equiv$ *or* $\Leftrightarrow$.

Two ways of proofs *for* ***logical equivalences***

1. algebraic ***rules*** *of logical equivalences*

2. ***truth tables*** *n propositional variables,* 2^n *raws in the table*

Precedence of Logical Operators

$\neg$ *high*

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

(=(≡, ⇔) *low*

*Disjunction($\vee$) and conjunction($\wedge$) are **associative**,*
Let $p_1, p_2, \ldots, p_n$ be n propositions. Then

$$\vee_{i=1}^{n} p_i = p_1 \vee p_2 \vee \ldots \vee p_n, \qquad \wedge_{i=1}^{n} p_i = p_1 \wedge p_2 \wedge \ldots \wedge p_n.$$

*Let I be a set(**index set**). Then we can define*

$\vee_{i \in I}\, p_i$ ***disjunctive normal form***

$\wedge_{i \in I}\, p_i$ ***conjunctive normal form***

See p8 of set equalities

Extened De Morgan's law

$$\neg \vee_{i \in I}\, p_i = \wedge_{i \in I} \neg p_i, \qquad \neg \wedge_{i \in I}\, p_i = \vee_{i \in I} \neg p_i.$$

$$\overline{\cup_{i \in I}}\, A_i = \cap_{i \in I}\, \overline{A}_i, \qquad \overline{\cap_{i \in I}}\, A_i = \cup_{i \in I}\, \overline{A}_i.$$

Logical equivalences involving conditional statements

$p \rightarrow q = \neg p \vee q$ *disjuntive normal form*

$\neg(p \rightarrow q) = p \wedge \neg q$ *conjuntive normal form*

$p \rightarrow q = \neg q \rightarrow \neg p$ *contrapositive*

$p \vee q = \neg p \rightarrow q$ $\quad p \wedge q = \neg(p \rightarrow \neg q)$

$(p \rightarrow q) \wedge (p \rightarrow r) = p \rightarrow (q \wedge r)$ $\quad (p \rightarrow q) \wedge (r \rightarrow q) = (p \vee r) \rightarrow q$

$(p \rightarrow q) \vee (p \rightarrow r) = p \rightarrow (q \vee r)$ $\quad (p \rightarrow q) \vee (r \rightarrow q) = (p \wedge r) \rightarrow q$

Set identities involving subset relations

$A \subseteq B = (\overline{A} \cup B = U)$ *disjuntive normal form*

$(\overline{A \subseteq B}) = (A \cap \overline{B} = U)$ *conjuntive normal form*

$A \subseteq B = \overline{B} \subseteq \overline{A}$ *contrapositive*

$A \cup B = \overline{A} \subseteq B$ $\quad A \cap B = \neg(A \subseteq \overline{B})$

$(A \subseteq B) \cap (A \subseteq C) = A \subseteq (B \cap C)$ $\quad (A \subseteq B) \cap (C \subseteq B) = (A \cup C) \subseteq B$

$(A \subseteq B) \cup (A \subseteq C) = A \subseteq (B \cup C)$ $\quad (A \subseteq B) \cup (C \subseteq B) = (A \cap C) \subseteq B$

Logical equivalences involving biconditional statements

$p \leftrightarrow q = q \leftrightarrow p$ commutative(symetric)

$p \leftrightarrow q = (p \rightarrow q) \wedge (q \rightarrow p)$ definition of biconditional

$p \leftrightarrow q = \neg p \leftrightarrow \neg q$ symetricity of biconditional

$p \leftrightarrow q = (p \wedge q) \vee (\neg p \wedge \neg q)$ disjuntive normal form(truth table)

$= (p \vee \neg q) \wedge (\neg p \vee q)$ conjuntive normal form

$\neg(p \leftrightarrow q) = (p \leftrightarrow \neg q) = (\neg p \leftrightarrow q)$ De Mogran's law for bicondi.

Set identities involving set equilities

$(A = B) = (B = A)$ commutative(symetric)

$(A = B) = (A \subseteq B) \wedge (B \subseteq A)$ definition of set equality

$(A = B) = (\overline{A} = \overline{B})$ symetricity of equality

$(A = B) = (A \wedge B) \vee (\overline{A} \wedge \overline{B})$ disjuntive normal form(Venn diagram)

$= (A \vee \overline{B}) \wedge (\overline{A} \vee B)$ conjuntive normal form

$\neg(A = B) = (A = \overline{B}) = (\overline{A} = B)$ De Mogran's law for set equa.

1.4 Arguments and Rules of Inference

Deductive(연역) reasoning

Hypotheses(Premises) $\Rightarrow$ ***Conclusion***

***Proof**:* ***valid arguments** that*

establish the truth of logical statements

arguments *a sequence of statement that ends with a **conclusion***

valid *the **conclusion** must follow from the truth of*

*the **preceding statements** or **premises of the argument***

*An **argument** is **valid**, if and only if,*

*it is impossible for **all premises** to be **true** and **conclusion** to be **false***

*or If **all premises** are **true**, then the conclusion is true.*

Rules of inference

***deducing** new statements from statements we already have.*

Valid Arguments in Propositional Logic

argument *a sequence of propositions with preceding* ***premises*** *and finally a* ***conclusion****.*

argument form

$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q$

valid *argument*

$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q$ *is* ***tautology****.*

$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \Rightarrow q$.

Def. 1.4.1

p_1

p_2

…

$\underline{p_n}$

q.

Fig. 1.4.1 Rules of Inference

Modus ponen

$$\begin{array}{l} p \to q \\ \underline{p \qquad\quad} \\ \therefore\ q \end{array}$$

Modus tollen

$$\begin{array}{l} p \to q \\ \underline{\neg q \qquad} \\ \therefore\ \neg p \end{array}$$

Addition

$$\begin{array}{l} \underline{p \qquad\qquad} \\ \therefore\ p \vee q \end{array}$$

Simplification

$$\begin{array}{l} \underline{p \wedge q} \\ \therefore\ p \end{array}$$

Conjunction

$$\begin{array}{l} p \\ \underline{q} \\ \therefore\ p \wedge q \end{array}$$

Hypothetical syllogism

$$\begin{array}{l} p \to q \\ \underline{q \to r} \\ \therefore\ p \to r \end{array}$$

Disjunction syllogism

$$\begin{array}{l} p \vee q \\ \underline{\neg p \quad} \\ \therefore\ q \end{array}$$

Resolution

$$\begin{array}{l} p \vee q \\ \underline{\neg p \vee r} \\ \therefore\ q \vee r \end{array}$$

Resolution

Automatic theorem prooving tools

***Useful** rule in reducing the size of propositions*

$p \lor q$	$p \lor q$	$p \lor q$
$\underline{\neg p \lor r}$	$\underline{\neg p \lor q}$	$\underline{\neg p}$
$\therefore\ q \lor r$	$\therefore\ q$	$\therefore\ q$
		disjunction syllogsm

Fallacies in Implication

$((p \to q) \land q) \not\Rightarrow p$ ***fallacy** of **affirming** the **conclusion***

***Implication** says **nothing** even though the **conclusin** is **true**!*

$((p \to q) \land \neg p) \not\Rightarrow \neg q$ ***fallacy** of **denying** the **hypothesis***

***Implication** says **nothing** when the **hypotheses** are **false**!*

1.5 Predicates and Quantifiers

Def. 1.5.1 predicate(propositional function; 조건명제) *P(x) is a **boolean function** with **variable** x in the **domain of discourse** D.*
*A **predicate** P(x) is either **T** or **F** for each d ∈ D.*

Exa. 1.5.2 *Let P(n) denotes "n is an odd integer". Then*

*P(3) denotes "3 is an odd integer" is **T**.* *proposition*
*P(4) denotes "4 is an odd integer" is **F**.* *proposition*
P(n) denotes "n is an odd integer" is ? ***not** a proposition*

Def. 1.5.1' predicate** P: D → {**T**, **F**} = **B**.* ***B**: **boolean
*for each **variable** x ∈ D: P(x) ∈ **B**.*

Quantifiers

A ***predicate*** *is* ***not*** *a* ***proposition*** *only if,* ***variables*** *are* ***not fixed****.*

If ***all*** *the variables are fixed, the predicate* ***becomes*** *a* ***propositions****.*

How can we fix variables?

Let ***universe(domain) of discourse*** *is D for each* ***predicates****.*

Let D be a set and $P: D \rightarrow \mathbf{B}$.

If P(x) is true for ***all*** *values of x in the* ***universe of discourse****,*

$\forall x \in D: P(x)$ *is true*

otherwise $\forall x P(x)$ *is false.*

$\therefore \forall x \in D: P(x)$ *becomes a* ***proposition****.*

Def. 1.5.4 Universal quantifier

"$\forall x \in D: P(x)$" (or "$\forall x\ P(x)$" for short) is a ***proposition*** *such that*
"P(x) (is ***T****) for* ***all*** *values in the* ***domain*** *D."*

$\forall$ is called ***universal*** *quantifier.*

We read "$\forall x \in D: P(x)$" as "for ***all*** *x in D, P(x) (is* ***T****)."*

Def. 1.5.9 Existential quantifier

"$\exists x \in D: P(x)$" (or "$\exists x\ P(x)$" for short) is a ***proposition*** *such that*
"There ***exists*** *an element x in D such that P(x) (is* ***T****)."*

$\exists$ is called ***existential*** *quantifier.*

We read "$\exists x \in D: P(x)$" as "there ***exists*** *x in D, P(x)".*

Let D be the ***universe of domains****. Then*

$\forall x \in D: P(x) = \wedge_{d \in D} P(d).$ $\exists x \in D: P(x) = \vee_{d \in D} P(d).$

function *"$\forall x \in D\ P(x)$" $\in$ **B*** ***(Exa. 1.5.7)***

for $d \in D$ ***do if*** $P(d) \to$ ***skip*** $\mid \neg P(d) \to$ ***return F fi od; return T***.

function *"$\exists x \in D\ P(x)$" $\in$ **B*** ***(Exa. 1.5.12)***

for $d \in D$ ***do if*** $P(d) \to$ ***return T*** $\mid \neg P(d) \to$ ***skip fi od; return F***.

pseudo code *of E. W. Dijkstra*

do-od *and* ***if-fi*** *bracket (), {}, []*

dangling-else probem in ***if-then-else*** *statement*

if B_1 ***then if*** B_2 ***then*** S_A ***else*** S_B.

for $x \in X$ ***do*** … ***od*** *structure*

if $B_1 \to \ldots \mid B_2 \to \ldots \mid \ldots \mid B_n \to \ldots$ ***fi***

Difficult *to proove $\forall x \in D: P(x)$, BUT An element*

$d \in D$.϶. $\neg P(d)$ is enough to proove $\neg \forall x \in D: P(x)$

counterexample *(반례) of $\forall x \in D: P(x)$.*

Binding variables

A variable is said to be ***bound****, if the variable binds to*

(1) quantifiers(∀, ∃) or

(2) specific value(in the domain), and

it is said to be ***free****, otherwise.*

A predicate with ***bound variables only*** *is a proposition but*

A peedicate with ***free varaible*** *is* ***not*** *a proposition*

Extension 1.5.1" predicate $P: D_1 \times D_2 \times \ldots \times D_n \to \mathbf{B}$.

for a n-tuple bariable $(x_1, x_2, \ldots, x_n) \in D_1 \times D_2 \times \ldots \times D_n$:

$$P(x_1, x_2, \ldots, x_n) \in \mathbf{B}.$$

scope *of* ***quantifier***

the ***part of logical expression*** *to which the quantifier is applied*

Example for ***scope*** *of* ***variables***

$$\exists x(P(x) \wedge R(x)) \vee \forall x R(x) \equiv \exists x(P(x) \wedge R(x)) \vee \forall y R(y)$$

Negation of Quantifiers

Let P(x) be "Every student in CS204 has taken CS101."

$\neg\forall x\, P(x)$

*"It is **not true** that **every** student in CS204 who has taken CS101."*

*is **logically equivalent** to*

*"**There is** a student in CS204 who has **not** taken CS101"*

$\exists x \neg P(x)$

$\therefore\ \neg\forall x\, P(x)) \equiv \exists x\, \neg P(x)$

$\neg\exists x\, P(x) \overset{?}{\equiv} \forall x \neg P(x)$

*"It is **not true** that **there's** a student in CS204 who has taken CS101."*

*is **logically equivalent** to*

*"**Every** student in CS204 has **not** taken CS101"*

Thm. 1.5.14 De Morgan's Laws for quantified predicates

$\neg\forall x P(x) = \exists x \neg P(x)$, $\neg\exists x P(x) = \forall x \neg P(x)$.

proof *Let D be a set of discourse. Then* *(diffrent form text)*

$\neg\forall x P(x) = \neg\wedge_{d\in D} P(d) = \vee_{d\in D} \neg P(d) = \exists x \neg P(x)$.

$\neg\exists x P(x) = \neg\vee_{d\in D} P(d) \equiv \wedge_{d\in D} \neg P(d) = \forall x \neg P(x)$.

Rules of Inferences

Universal instatiation

$\underline{\forall x \in D: P(x)}$

$\therefore P(d)$, *if* $d \in D$

Universal generalization

$\underline{P(d) \text{ for every } d \in D}$

$\therefore \forall x P(x)$

Exsistential instatiation

$\underline{\exists x \in D: P(x)}$

$\therefore P(d)$ *for some* $d \in D$

Exsistential generalization

$\underline{P(d) \text{ for some } d \in D}$

$\therefore \exists x P(x)$

1.6 Nested Quantifier

$\forall x \forall y\ P(x, y)$

function "$\forall x{\in}D\forall y{\in}E$: $P(x, y)$" $\in$ ***B*** $\forall x \forall y\ P(x, y)$

```
for d ∈ D do
    for e ∈ E do
        if P(d, e) → skip | ¬P(d, e) → return F fi
        od
    od;
return T
```

function "$\exists x{\in}D\exists y \in E$: $P(x, y)$" $\in$ ***B*** $\exists x \exists y\ P(x, y)$

```
for (d, e) ∈ D × E do
    if P(d, e) → return T | ¬P(d, e) → skip fi
    od;
return F
```

```
function "∀x∈ D∃y∈E: P(x, y)" ∈ B                          ∀x∃y P(x, y)
    for d ∈ D do                                             ∀x(d)
        for e ∈ E do                                             ∃y(e)
            if P(d, e) → break | ¬P(d, e) → skip fi
        od;
        if P(d, e) → skip |  e ∉ E → return F fi
    od;
    return T
function"∃x∈D∀y∈E: P(x, y)" ∈ B                            ∃x∀y P(x, y)
    for d ∈ D do                                             ∃x(d)
        for e ∈ E do                                             ∀y(e)
            if P(d, e) → skip | ¬P(d, e) → break fi
        od;
        if e ∉ E → return T | ¬P(d, e) → skip fi
    od;
    return F
```

function *"$\forall x\in D\exists y\in E$: $P(x, y)$" $\in$ **B*** $\qquad$ $\forall x\exists y\, P(x, y)$

$\quad$ $(d, \forall)$:= (*first*(**D**), **F**); ***while*** $\neg\forall$ ***do*** $\qquad$ $\forall x(\mathbf{d})$

$\qquad$ $(e, \exists)$:= (*first*(**E**), **F**); ***while*** $\neg\exists$ ***do*** $\qquad$ $\exists y(\mathbf{e})$

$\qquad\quad$ ***if*** $P(d, \mathbf{e}) \rightarrow \exists$:= **T** | $\neg P(d, \mathbf{e}) \rightarrow (e, \exists)$:= $next^2(\mathbf{E})$ ***fi***

$\qquad$ ***od;***

$\qquad$ ***if*** $\exists \rightarrow (d, \forall)$:= *next*(**D**) | $\neg\exists \rightarrow$ ***return*** **F** ***fi***

$\quad$ ***od;***

$\quad$ ***return T***

function *"$\exists x\in D\forall y\in E$: $P(x, y)$" $\in$ **B*** $\qquad$ $\exists x\forall y\, P(x, y)$

$\quad$ $(e, \exists)$:= (*first*(**D**), **F**); ***while*** $\neg\exists$ ***do*** $\qquad$ $\exists x(\mathbf{d})$

$\qquad$ $(d, \forall)$:= (*first*(**E**), **F**); ***while*** $\neg\forall$ ***do*** $\qquad$ $\forall y(\mathbf{e})$

$\qquad\quad$ ***if*** $P(d, \mathbf{e}) \rightarrow (e, \exists)$:= $next^2(\mathbf{E})$ | $\neg P(d, \mathbf{e}) \rightarrow \exists$:= **F** ***fi***

$\qquad$ ***od;***

$\qquad$ ***if*** $\forall \rightarrow$ ***return*** **T** | $\neg\forall \rightarrow (d, \exists)$:= *next*(**D**) ***fi***

$\quad$ ***od;***

$\quad$ ***return F***

function *"∀x∈ D∃y∈E: P(x, y)"* ∈ ***B*** $\forall x \exists y\, P(x, y)$

(d, ∀) **:=** *(first(***D***),* **F***);* **do** ¬∀ → $\forall x(\mathbf{d})$

(e, ∃) **:=** *(first(***E***),* **F***);* **do** ¬∃ → $\exists y(\mathbf{e})$

if *P(d,* ***e****)* → ∃ **:= T** | ¬*P(d,* ***e****)* → *(e, ∃)* **:=** *next(***E***)* ***fi***

od;

if ∃ → *(d, ∀)* **:=** *next²(***D***)* | ¬∃ → ***return*** **F** ***fi***

od;

return T

function *"∃x∈D∀y∈E: P(x, y)"* ∈ ***B*** $\exists x \forall y\, P(x, y)$

(e, ∃) **:=** *(first(***D***),* **F***);* **do** ¬∃ → $\exists x(\mathbf{d})$

(d, ∀) **:=** *(first(***E***),* **F***);* **do** ¬∀ → $\forall y(\mathbf{e})$

if *P(d,* ***e****)* → *(e, ∃)* **:=** *next²(***E***)* | ¬*P(d,* ***e****)* → ∃ **:= F** ***fi***

od;

if ∀ → ***return*** **T** | ¬∀ → *(d, ∃)* **:=** *next(***D***)* ***fi***

od;

return F