Chapter 2. 논리와 명제

개요

- 논리와 명제와 관련된 전반적인 논제들을 고찰함
- 명제를 정의하고, 단순 명제와 합성 명제를 통한 논리 연산을 다룸
- 6가지 주요 논리 연산자, 즉 부정, 논리합, 논리곱, 배타적 논리합, 조건, 쌍방 조건들을 진리표를 통하여 살펴봄
- 어떤 명제의 역, 이, 대우를 고찰하며, 항진 명제와 모순 명제를 정의함
- 논리적 동치 관계와 주요 성질들을 고찰함
- 추론을 정의하여 그것과 관련된 몇 가지 예를 살펴보며 전체 한정자와 존재 한정자를 다룸

CONTENTS

- **논리**

  인간의 사고가 논리적인지를 판단하는 것은 사고하는 사람이 주어진 문제를 객관적으로 명확한지의 여부와 사고의 법칙을 체계적으로 추구하여 분석하는지의 여부로 결정됨

- **논리의 목적**
  - 특정한 논리를 통한 입증이 옳은가를 측정하는 데 필요한 법칙을 제공한다는 점
  - 논리에 대한 연구와 응용 분야임
  - 컴퓨터 관련 학문이나 공학 등 여러 분야에 폭넓게 응용됨
    알고리즘의 설계나 증명, 디지털 논리 회로의 설계, 논리 프로그램 관련 분야, 관계형 데이터베이스 이론, 오토마타와 계산 이론, 인공지능 등에 필요한 이론적 기반을 제공함

- **명제 논리(Propositional Logic)**
  - 주어와 술어를 구분하지 않고 전체를 하나의 식으로 처리하여 참 또는 거짓을 판별하는 법칙임
- **술어 논리(Predicate Logic)**
  - 주어와 술어로 구분하여 참 또는 거짓에 관한 법칙임

2.1 논리와 명제

정의 2-1 명제(proposition)란 어떤 사고를 나타내는 문장 중에서 참(true)이나 거짓(false)을 객관적이고 명확하게 구분할 수 있는 문장이나 수학적 식을 말한다.

- 명제는 통상 영문 소문자 p, q, r, $\cdots$ 등으로 표기함
- 명제가 참 또는 거짓의 값을 가질 때 그 값을 명제의 진리값(truth value)이라고 함
- 명제의 진리 값은 참일 때는 T(true), 거짓일 때는 F(false)로 각각 표시함
- 명제는 T와 F의 2가지의 진리 값만을 가지므로 이진 논리라고 함

여기서 잠깐!!

명제란 어떤 문장이나 식이 애매하지 않고, 참과 거짓이 명백해야 한다. 예를 들어, '$3 + 2 = 4$'는 거짓인 문장이다. 그러나 '지금 몇 시인가?' 또는 '과제물을 가급적 빨리 제출하시오' 등의 문장은 그것이 참인지 거짓인지 알 수 없기 때문에 명제가 아니다.

예제 ②-1 다음의 문장이나 식에서 명제를 찾아보고, 명제인 경우 그것의 진리값을 판별해보자.

(1) 바나나는 맛있다.

(2) $3x + 5y = 7$

(3) 28은 4의 배수이다.

(4) 지금 어디로 가는 중입니까?

풀이 (1), (2), (4)는 참이나 거짓을 판별할 수 없으므로 명제가 될 수 없다. (3)은 참이기 때문에 명제이고 진리값은 T이다.

예제 2-2 다음의 문장들은 모두 명제이다. 이 명제들의 참, 거짓을 판별해보자.

(1) $6 < 4$

(2) 유채꽃은 노란색이다.

(3) 3×7의 값은 홀수이다.

(4) 공기는 H_2O로 표현된다.

풀이 (1)과 (4)는 거짓이고, (2)와 (3)은 참이다.

정의 2−2 하나의 문장이나 식으로 구성되어 있는 명제를 단순 명제(simple proposition)라 하고, 여러 개의 단순 명제들이 논리 연산자들로 연결되어 만들어진 명제를 합성 명제(composition proposition)라고 한다.

예를 들어, '장미꽃은 빨갛다'와 '유채꽃은 노랗다'는 각각 단순 명제이고, '장미꽃은 빨갛고 유채꽃은 노랗다'는 합성 명제이다.

- **논리 연산자(Logical Operators)**

  단순명제들을 연결시켜 주는 역할을 하는 ∨, ∧, ~과 같은 연결자라고 함

- **합성 명제의 진리 값**
  - 그 명제를 구성하는 단순 명제의 진리 값과 논리 연산자의 특성에 따라 값이 정해짐
  - 단순 명제의 진리 값은 그 명제가 참이냐 거짓이냐에 따라 T 또는 F로 표시함
  - 합성 명제의 진리 값은 복잡한 경우가 많음
  - 진리 표(truth table)를 사용하여 단계적으로 연산함으로써 원하는 합성 명제의 진리 값을 보다 쉽고 편리하게 구할 수 있음

2.2 논리 연산

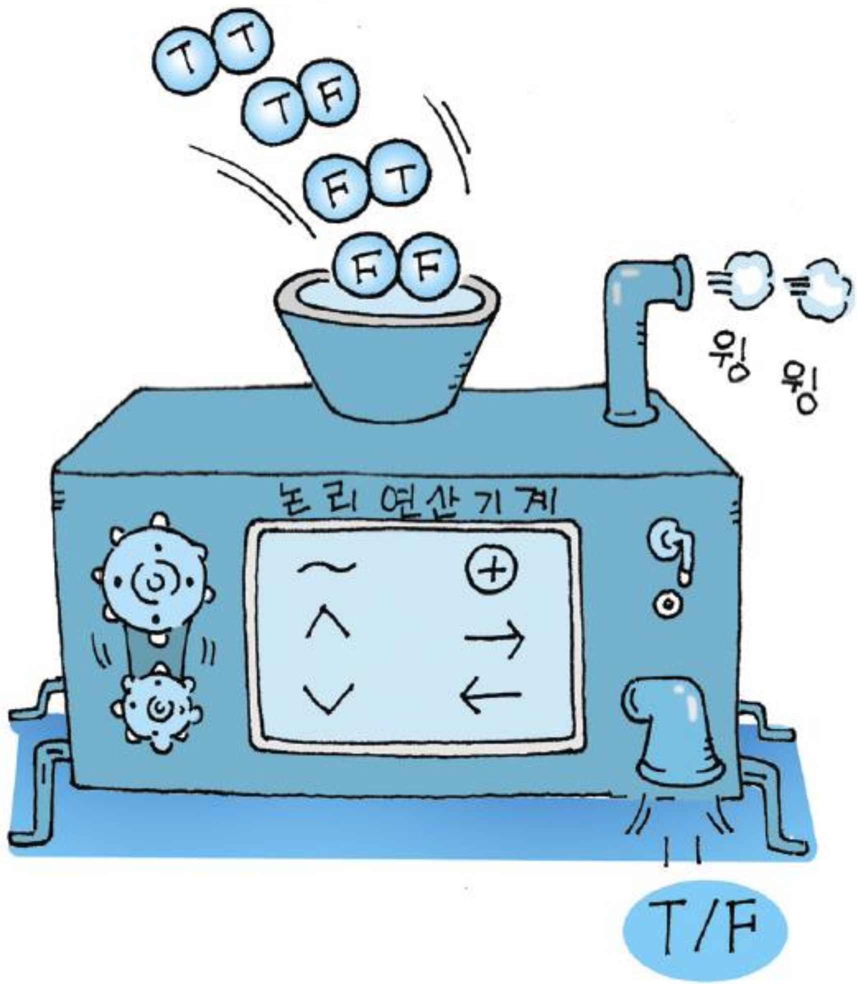
여기서 잠깐!!

논리 기호 ∨, ∧, ~ 등을 여러 번 사용하여 합성 명제를 구성할 경우 합성 명제 $P(p, q, r \cdots)$의 부분 명제 $p, q, r \cdots$ 등을 **변수**라 하고, 합성 명제의 진리값은 각 변수들의 진리값에 의해 구할 수 있다.

〈표 2.1〉 논리 연산자의 이름과 기호

연산자의 이름	기호	연산자의 의미
부정	$\sim$	NOT
논리곱	$\wedge$	AND
논리합	$\vee$	OR
배타적 논리합	$\oplus$	Exclusive OR
조건	$\rightarrow$	if … then
쌍방 조건	$\leftrightarrow$	if and only if (iff)

2.2 논리 연산



(1) 부정(Negation)

- 임의의 명제 p가 주어졌을 때 그 명제에 대한 부정(negation)은 명제 p의 반대되는 진리값을 가짐
  - 기호로는 $\sim p$라 쓰고 'not p 또는 p가 아니다' 라고 함
  - p의 진리값이 참이면 $\sim p$의 진리 값은 거짓임
  - p의 진리값이 거짓이면 $\sim p$의 진리 값은 참임

예제 2-4 p가 명제일 때 진리표를 이용하여 $\sim(\sim p)$의 진리값이 p의 진리값과 같음을 보이자.

풀이 p와 $\sim(\sim p)$에 대한 진리표는 다음과 같다.
따라서 p와 $\sim(\sim p)$의 진리값은 같다.

p	$\sim p$	$\sim(\sim p)$
T	F	T
F	T	F

같은 값

(2) 논리곱(Conjunction)

- 임의의 두 명제 p, q가 '그리고(AND)' 로 연결되어 있을 때 명제 p, q의 은 $p \wedge q$로 표시함
- ' p *and* q 또는 p 그리고 q' 라고 함
- 두 명제의 논리곱 $p \wedge q$는 두 명제가 모두 참인 경우에만 참이라고 함
- 그렇지 않으면 거짓의 진리 값을 가짐

〈표 2.3〉 논리곱에 대한 진리표

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

예제 ❷-5 다음의 합성 명제를 단순 명제들로 분리하여 단순 명제의 진리값을 구하고, 앞의 진리표를 이용하여 논리곱으로 이루어진 합성 명제의 진리값을 구해보자.

(1) 서울은 대한민국의 수도이고, 런던은 영국의 수도이다.

(2) $3 > 2$이고, $3 \times 2 = 5$이다.

(3) 사과는 과일이고, 시금치는 채소이다.

풀이 (1) '서울은 대한민국의 수도이다'와 '런던은 영국의 수도이다'가 된다. 두 단순 명제의 진리값이 모두 T이므로 주어진 합성 명제의 진리값도 T이다.

(2) '$3 > 2$'와 '$3 \times 2 = 5$'이다. 각 단순 명제의 진리값은 T와 F이므로 합성 명제의 진리값은 F이다.

(3) '사과는 과일이다'와 '시금치는 채소이다'이다. 이 명제들의 진리값은 모두 T이므로 합성 명제의 진리값 또한 T이다.

(3) 논리합(Disjunction)

- 임의의 두 명제 p, q가 '또는(OR)'으로 연결되어 있을 때 명제 p, q의 은 $p \lor q$로 표시함
- 'p or q나 p 또는 q' 라고 표현함
- 두 명제의 논리합 $p \lor q$는 두 명제가 모두 거짓인 경우에만 거짓의 진리값을 가짐

〈표 2.4〉 논리합에 대한 진리표

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

예제 2-6 다음의 합성 명제를 단순 명제로 구분하여 단순 명제의 진리값을 구하고, 앞의 진리표를 사용하여 논리합으로 이루어진 합성 명제의 진리값을 구해보자.

(1) 서울은 대한민국의 수도이거나, 런던은 영국의 수도이다.

(2) $3 > 2$ 또는 $3 \times 2 = 5$이다.

(3) 사과는 과일이거나, 시금치는 채소이다.

풀이 (1) 두 단순 명제의 진리값이 각각 T이므로 합성 명제의 진리값도 T이다.

(2) 한 단순 명제는 T이고 다른 단순 명제는 F이므로 합성 명제는 T이다.

(3) 두 단순 명제의 진리값은 T이므로 합성 명제의 진리값도 T이다.

(4) 배타적 논리합(Exclusive OR)

- 임의의 두 명제 p, q의 $p \oplus q$로 표시함
- ‘익스클루시브 OR(Exclusive OR) 또는 XOR’이라고 읽음
- 진리 값은 p와 q 두 명제 중에서 하나의 명제가 참이고 다른 하나의 명제가 거짓일 때 참의 진리 값을 가지고, 그렇지 않으면 거짓의 진리 값을 가짐

〈표 2.5〉 배타적 논리합에 대한 진리표

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

(5) 조건(Implication)

- 조건 연산자를 함축(implication)이라고 함
- 임의의 명제 p, q의 조건 연산자는 $p \rightarrow q$로 표시함
- 'p이면 q이다' 라고 읽음

〈표 2.6〉 조건에 대한 진리표

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

조건 연산자 → 는 'p이면 q이다' 뿐만 아니라, 다음과 같이 똑같은 진리 값을 가지는 다양한 표현으로 나타내질 수 있음

(a) p이면 q이다. (if p, then q)

(b) p는 q의 충분조건이다. (p is sufficient for q)

(c) q는 p의 필요조건이다. (q is necessary for p)

(d) p는 q를 함축한다. (p implies q)

여기서 잠깐!!

$$\underset{(\text{충분조건})}{p} \rightarrow \underset{(\text{필요조건})}{q}$$

예제 ②-7

다음의 명제에 대하여 함축의 진리값을 구해보자.

(1) 바다가 육지라면, 런던은 영국의 수도이다.

(2) $3+4>5$이면, $3>5$이다.

(3) 유채꽃이 빨갛다면, 바다가 육지이다.

풀이 (1) p : '바다가 육지이다' 는 F이고, q : '런던은 영국의 수도이다' 는 T이다. 그러므로 명제 $p \rightarrow q$의 진리값은 T이다.

(2) p : '$3+4>5$' 는 T이고, q : '$3>5$' 는 F이다. 그러므로 명제 $p \rightarrow q$의 진리값은 F이다.

(3) p : '유채꽃이 빨갛다' 는 F이고, q : '바다가 육지이다' 는 F이다. 그러므로 합성 명제 $p \rightarrow q$의 진리값은 T이다.

(6) 쌍방 조건

- 임의의 명제 p, q의 쌍방 조건은 $p \leftrightarrow q$로 표시함
- 'p이면 q이고, q이면 p이다' 라고 함
- 진리 값은 p, q가 모두 참이거나 거짓일 때 참의 값을 가지고, 그 외에는 거짓의 값을 가짐

〈표 2.7〉 쌍방 조건에 대한 진리표

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

쌍방 조건도 같은 의미를 가진 다른 표현으로 나타내질 수 있음

(1) p이면 q이고, q이면 p이다. (p if and only if q)

(2) p는 q의 필요충분조건이다. (p is necessary and sufficient for q)

여기서 잠깐!!

합성 명제를 구성하는 데 있어서 여러 개의 논리 연산자들이 쓰일 수도 있다. 그러나 여러 개의 논리 연산자를 포함하는 합성 명제에서 괄호가 없는 경우에는 다음과 같은 우선순위에 의해 실행된다.

$\sim \; > \; \wedge \; > \; \vee \; > \; \rightarrow \; > \; \leftrightarrow$

예제 ②-8 합성 명제 $\sim(p \wedge \sim q)$의 진리값을 구해보자.

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

위의 표에서와 같이 변수가 2개일 경우에는 4개의 행이 필요함을 알 수 있다. 그리고 각 진리값들은 앞에서 정의한 $\wedge$, $\vee$, $\sim$ 등의 연산을 순서에 따라 적용하여 구할 수 있으며, 마지막 열에 있는 것이 구하고자 하는 합성 명제의 진리값이다.

예제 ②-9 p, q, r이 명제일 때 다음의 합성 명제에 대한 진리표를 만들어보자.

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

풀이 변수가 3개일 때는 2^3, 즉 8개의 행이 필요함을 알 수 있다. 이 경우에 T와 F의 조합을 다음의 표와 같은 순서로 나타내는 것이 효율적이다. 여기서는 $q \wedge r$ 을 먼저 연산하고, $p \vee (q \wedge r)$을 그 후에 연산한다.

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

예제 ②-10 명제 p를 '날씨가 춥다' 그리고 명제 q를 '비가 온다'라고 할 때 다음 각각의 명제들을 문장으로 표현해보자.

(1) $\sim p$ (2) $p \wedge q$ (3) $p \vee q$ (4) $p \vee \sim q$

풀이 각 경우에 있어 $\wedge$, $\vee$, $\sim$을 각각 '그리고, 또는, 아니다'란 우리말로 바꾸면 된다.

(1) 날씨가 춥지 않다. (2) 날씨가 춥고 비가 온다.

(3) 날씨가 춥거나 비가 온다. (4) 날씨가 춥거나 비가 오지 않는다.

예제 ②-11 다음 합성 명제를 단순 명제들로 구분하고, 진리표를 이용하여 합성 명제의 진리값을 구해보자.

(1) $4+3=7$이고, $4\times7=9$이다.

(2) 유채꽃이 노랗다면, 수요일 전날은 토요일이다.

풀이 (1) p : '$4+3=7$' 은 참이고 q : '$4\times7=9$'는 거짓이다. 따라서 합성 명제 $p\wedge q$는 거짓이다.

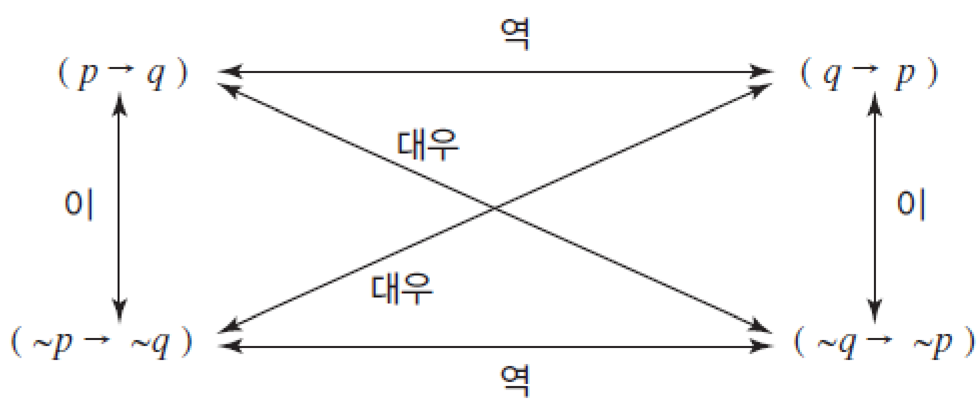
(2) p : '유채꽃이 노랗다' 는 참이고, q : '수요일 전날은 토요일이다' 는 거짓이다. 따라서 합성 명제 $p\rightarrow q$는 거짓이다.

명제 $p \rightarrow q$에 대하여

$q \rightarrow p$를 역(converse)

$\sim p \rightarrow \sim q$를 이(inverse)

$\sim q \rightarrow \sim p$를 대우(contrapositive)



(명제의 역, 이, 대우의 상호 관계)

〈표 2.8〉 역, 이, 대우 간의 관계에 대한 진리표

				(명제)	(역)	(이)	(대 우)
p	q	$\sim p$	$\sim q$	$p \rightarrow q$	$q \rightarrow p$	$\sim p \rightarrow \sim q$	$\sim q \rightarrow \sim p$
T	T	F	F	T	T	T	T
T	F	F	T	F	T	T	F
F	T	T	F	T	F	F	T
F	F	T	T	T	T	T	T

같은 값

같은 값

예제 2-12 명제 $p \rightarrow q$: '날씨가 맑아지면 소풍을 간다'의 역, 이, 대우를 구해보자.

풀이 역($q \rightarrow p$) : '소풍을 가면 날씨가 맑아진다'

이($\sim p \rightarrow \sim q$) : '날씨가 맑아지지 않으면 소풍을 가지 않는다'

대우($\sim q \rightarrow \sim p$) : '소풍을 가지 않으면 날씨가 맑아지지 않는다'

여기서 잠깐!!

논리란 인간이 어떻게 사고하는가를 표현하는 사고의 규칙을 말한다. 논리에 관한 최초의 논문은 그리스 철학자 아리스토텔레스(Aristoteles)에 의해 쓰여졌는데, 모든 분야의 지식 연구에 바탕이 되는 연역적 추론을 위한 규칙들로 채워져 있다. 17세기에 독일의 수학자 라이프니츠(Leibniz)는 연역 추론 과정을 명확히 하기 위해 기호를 사용하는 것을 고안해 내었다. 그가 생각한 개념은 19세기에 들어와서 현대 기호 논리의 창시자인 미국의 수학자 부울(Boole)과 드모르간(De Morgan)에 의해 확립되었다.

2.3 항진 명제와 모순 명제

정의 2-3 합성 명제에서 그 명제를 구성하는 단순 명제들의 진리값에 관계없이 그 합성 명제의 진리값이 항상 참(T)의 값을 가질 때 그 명제를 항진 명제(tautology)라고 한다.

정의 2-4 합성 명제에서 그 명제를 구성하는 단순 명제들의 진리값에 관계없이 그 합성 명제의 진리값이 항상 거짓(F)의 값을 가질 때 그 명제를 모순 명제(contradiction)라고 한다.

예제 2-13 p가 단순 명제일 때 $p \vee (\sim p)$는 항진 명제이고, $p \wedge (\sim p)$는 모순 명제임을 보이자.

풀이 합성 명제 $p \vee (\sim p)$와 $p \wedge (\sim p)$에 대한 진리표를 나타내면 다음과 같다.

p	$\sim p$	$p \vee (\sim p)$	$p \wedge (\sim p)$
T	F	T	F
F	T	T	F

$p \vee (\sim p)$의 진리값은 항상 참이므로 항진 명제이고, $p \wedge (\sim p)$의 진리값은 항상 거짓이므로 모순 명제이다.

여기서 항진 명제의 부정은 모순 명제임을 알 수 있고, 모순 명제의 부정은 항진 명제임을 알 수 있다.

예제 2-14 $(p \wedge q) \wedge \sim(p \vee q)$가 모순 명제임을 보이자.

풀이 진리표를 만들었을 때 $(p \wedge q) \wedge \sim(p \vee q)$의 진리표가 p와 q의 모든 값에 대해서 거짓이므로 모순 명제이다.

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

2.4 논리적 동치 관계

정의 2-5 두 개의 명제 p, q의 쌍방 조건 $p \leftrightarrow q$가 항진 명제이면, 두 명제 p, q는 논리적 동치(logical equivalence)라 하고, $p \equiv q$ 또는 $p \Leftrightarrow q$라고 표시한다. 즉, 명제 p와 q는 같은 논리값을 가진다는 의미이다.

- 두 명제가 논리적 동치일 경우는 두 명제의 논리값이 서로 같으므로 하나의 명제가 다른 명제를 대신할 수 있음
- 어떤 복잡한 명제를 좀 더 간단한 명제로 만들기 위해 논리적 동치 관계인 다른 명제를 사용하여 간소화함

2.4 논리적 동치 관계

예제 2-16 명제 $\sim(p \vee q)$와 $(\sim p) \wedge (\sim q)$이 논리적 동치임을 확인해보자.

풀이 두 명제 $\sim(p \vee q)$와 $(\sim p) \wedge (\sim q)$에 대한 진리값을 구하고, 서로의 진리값이 같음을 보이면 된다. 두 명제에 대한 진리표를 구하면 다음과 같다.

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

같은 값

위의 진리표에서 명제 $\sim(p \vee q)$의 진리값과 명제 $(\sim p) \wedge (\sim q)$의 진리값이 같으므로 $\sim(p \vee q) \Leftrightarrow (\sim p) \wedge (\sim q)$이다.

2.4 논리적 동치 관계

〈표 2.9〉 논리적 동치 관계의 기본 법칙

논리적 동치 관계	법칙 이름
$p \lor p \Leftrightarrow p$ $p \land p \Leftrightarrow p$	멱등 법칙 (idempotent law)
$p \lor \mathrm{T} \Leftrightarrow \mathrm{T}$ $p \lor \mathrm{F} \Leftrightarrow p$ $p \land \mathrm{T} \Leftrightarrow p$ $p \land \mathrm{F} \Leftrightarrow \mathrm{F}$	항등 법칙 (identity law)
$\sim\mathrm{T} \Leftrightarrow \mathrm{F}$ $\sim\mathrm{F} \Leftrightarrow \mathrm{T}$ $p \lor (\sim p) \Leftrightarrow \mathrm{T}$ $p \land (\sim p) \Leftrightarrow \mathrm{F}$	부정 법칙 (negation law)
$\sim(\sim p) \Leftrightarrow p$	이중 부정 법칙 (double negation law)

2.4 논리적 동치 관계

$p \vee q \Leftrightarrow q \vee p$ $p \wedge q \Leftrightarrow q \wedge p$ $p \leftrightarrow q \Leftrightarrow q \leftrightarrow p$	교환 법칙 (commutative law)
$(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$ $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$	결합 법칙 (associative law)
$p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$	분배 법칙 (distributive law)
$p \vee (p \wedge q) \Leftrightarrow p$ $p \wedge (p \vee q) \Leftrightarrow p$	흡수 법칙 (absorption law)
$\sim(p \vee q) \Leftrightarrow (\sim p) \wedge (\sim q)$ $\sim(p \wedge q) \Leftrightarrow (\sim p) \vee (\sim q)$	드 모르간 법칙 (De Morgan's law)
$p \rightarrow q \Leftrightarrow \sim p \vee q$	조건 법칙
$p \rightarrow q \Leftrightarrow \sim q \rightarrow \sim p$	대우 법칙

2.4 논리적 동치 관계

- **두 명제가 논리적 동치 관계임을 입증하는 방법**
  - 두 명제에 대한 진리표를 구하고 두 명제의 진리 값이 같음을 증명함
  - 하나의 명제로부터 논리적 동치 관계의 기본 법칙을 이용하여 다른 명제로 유도해 냄
  - 이 중 문제에 적당한 방법을 선택하여 문제를 해결함

예제 ②-17 드 모르간의 법칙을 이용하여 다음을 간단히 해보자.

(1) $\sim(p \wedge \sim q)$

(2) $\sim(\sim p \wedge \sim q)$

(3) $\sim(\sim p \vee q)$

풀이 (1) $\sim(p \wedge \sim q) \Leftrightarrow \sim p \vee \sim(\sim q)$

$\Leftrightarrow \sim p \vee q$

(2) $\sim(\sim p \wedge \sim q) \Leftrightarrow \sim(\sim p) \vee \sim(\sim q)$

$\Leftrightarrow p \vee q$

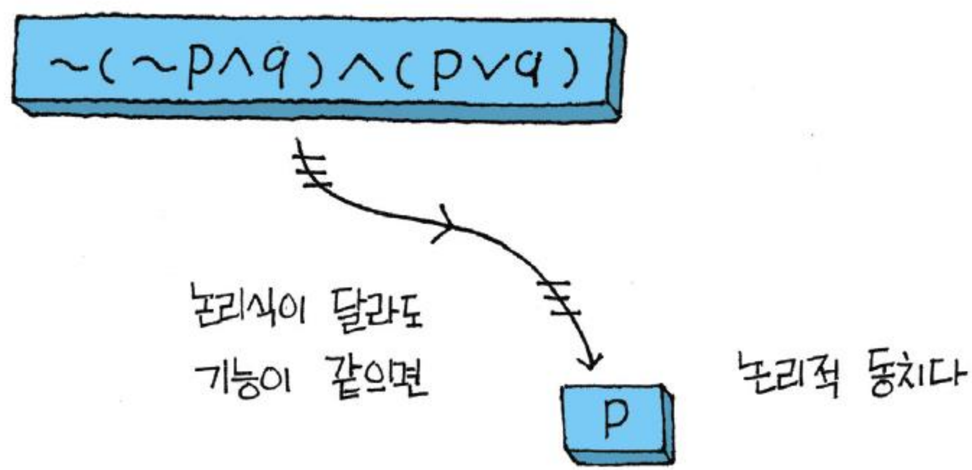
(3) $\sim(\sim p \vee q) \Leftrightarrow \sim(\sim p) \wedge \sim q$

$\Leftrightarrow p \wedge \sim q$

예제 ❷-18 논리적 동치 관계의 기본 법칙들을 이용하여 $\sim(\sim p \wedge q) \wedge (p \vee q) \equiv p$임을 보이자.

풀이

$$\begin{aligned}
\sim(\sim p \wedge q) \wedge (p \vee q) &\equiv (\sim(\sim p) \vee \sim q) \wedge (p \vee q) && \text{: 드 모르간의 법칙} \\
&\equiv (p \vee \sim q) \wedge (p \vee q) && \text{: 이중 부정 법칙} \\
&\equiv p \vee (\sim q \wedge q) && \text{: 분배 법칙} \\
&\equiv p \vee \mathrm{F} && \text{: 부정 법칙} \\
&\equiv p && \text{: 항등 법칙}
\end{aligned}$$



예제 2-19

쌍방 조건 $p \leftrightarrow q$는 'p이면 q이고, q이면 p이다' 이므로, 이것을 p, q 명제와 연산자로 표시하면 $(p \to q) \land (q \to p)$와 같다. 따라서 $p \leftrightarrow q$의 진리값과 $(p \to q) \land (q \to p)$의 진리값이 같음을 살펴보자.

풀이 이것을 진리표로 만들면 서로 같음을 알 수 있다.

p	q	$p \leftrightarrow q$	$p \to q$	$q \to p$	$(p \to q) \land (q \to p)$
T	T	T	T	T	T
T	F	F	F	T	F
F	T	F	T	F	F
F	F	T	T	T	T

같은 값

❖ 추론(Argument)

- 주어진 명제가 참인 것을 바탕으로 새로운 명제가 참이 되는 것을 유도해내는 방법임
- 주어진 명제들인 $p_1, p_2, \dots p_n$ 을 전제(premise)라고 함
- 새로이 유도된 명제 q를 결론(conclusion)이라고 함
- 유효 추론(valid argument)

  주어진 전제가 참이고 결론도 참인 추론
- 허위 추론(fallacious argument)

  추론의 결론이 거짓

〈표 2.10〉 여러 가지 추론 법칙

추론 법칙	법칙 이름
p $p \to q$ $\therefore q$	긍정 법칙 (modus ponens)
$\sim q$ $p \to q$ $\therefore \sim p$	부정 법칙 (modus tollens)
$p \to q$ $q \to r$ $\therefore p \to r$	조건 삼단 법칙 (hypothetical syllogism)
$p \vee q$ $\sim p$ $\therefore q$	선언 삼단 법칙 (disjunctive dilemma)

$(p \to q) \wedge (r \to s)$ $p \vee r$ $\therefore (q \vee s)$	양도 법칙 (constructive dilemma)
$(p \to q) \wedge (r \to s)$ $\sim q \vee \sim s$ $\therefore \sim p \vee \sim r$	파괴적 법칙 (destructive dilemma)
p $\therefore p \vee q$	선접 법칙 (disjunctive addition)
$p \wedge q$ $\therefore p$	분리 법칙 (simplication)
p q $\therefore p \wedge q$	연접 법칙 (conjunction)

예제 2-20 다음 $p \to q$, $p \vdash q$ 추론식에 나타난 명제들을 예를 들어 설명해보자.

풀이 위의 식에서 사용된 명제는 p, q 두 개이므로 p, q에 대한 예를

p : '오늘은 비가 온다'

q : '나는 공부를 한다'

라고 가정하면 추론식을 다음과 같이 표현한다.

'오늘 비가 오면 나는 공부를 한다'

'오늘은 비가 온다'

'그러므로 나는 공부를 한다'

예제 2-21 다음 추론이 유효 추론인지 허위 추론인지를 결정해보자.

$$p \to q,\ q \vdash p$$

풀이 추론 $p \to q,\ q \vdash p$에 대한 진리표를 만들면 다음과 같다.

	p	q	$p \to q$
$\Rightarrow$	T	T	T
	T	F	F
$\Rightarrow$	F	T	T
	F	F	T

진리표에서 전제 $p \to q$와 q가 모두 참인 경우는 $\Rightarrow$로 표시된 첫 번째와 세 번째 행이다. 두 경우 모두 추론의 결론인 p의 진리값을 살펴보면 첫 번째 행은 참이고 세 번째 행은 거짓의 진리값을 가진다. 그러므로 이 추론은 허위 추론이다.

가장 많이 사용되고 잘 알려진 3가지 논리 법칙

(a) 긍정 법칙 : $p,\ p \rightarrow q \vdash q$

(b) 부정 법칙 : $\sim q,\ p \rightarrow q \vdash \sim p$

(c) 삼단 법칙 : $p \rightarrow q,\ q \rightarrow r \vdash p \rightarrow r$

예제 2-22 [긍정 법칙] $p, p \rightarrow q \vdash q$가 유효 추론임을 진리표를 이용하여 보이자.

풀이 만들어진 진리표의 1행에서 p도 참이고 $p \rightarrow q$가 참인 경우(⇒ 마크한 부분)를 살펴보면, 결론인 q도 참(T)이므로 유효 추론이다.

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

예제 ②-23 [삼단 법칙] $p \rightarrow q,\ q \rightarrow r \vdash p \rightarrow r$ 이 유효 추론임을 진리표를 이용하여 보이자.

풀이 만들어진 진리표의 1, 5, 7, 8 행에서(⇒ 마크한 부분) $p \rightarrow q$와 $q \rightarrow r$ 모두 참(T)인 경우에, 결론인 $p \rightarrow r$도 모두 참이므로 삼단 법칙에 대한 추론은 유효 추론이다.

	p	q	r	$p \rightarrow q$	$q \rightarrow r$	$p \rightarrow r$
⇒	T	T	T	T	T	T
	T	T	F	T	F	F
	T	F	T	F	T	T
	T	F	F	F	T	F
⇒	F	T	T	T	T	T
	F	T	F	T	F	T
⇒	F	F	T	T	T	T
⇒	F	F	F	T	T	T

- 명제 중에는 값이 정해지지 않는 변수나 객체(object)가 있어서 참과 거짓을 판별하기 힘든 경우가 있음
- 변수의 값에 따라 그 명제가 참이 되고 거짓이 될 수 있음

"$x^2 + 5x + 6 = 0$이라는 명제는 x의 값이 -2 또는 -3일 경우에는 참의 값을 가지고 그 외에는 거짓의 값을 가진다. 이런 경우 우리는 $x^2 + 5\chi + 6 = 0$ 을 만족시키는 변수가 있다' 고 표현한다."

- 이와 같은 형태의 명제를 $p(x)$로 표시하고, $p(x)$를 변수 x 에 대한 명제 술어(propositional predicate)라고 함
- 명제 논리와 구분하여 명제 술어에 대한 논리를 술어 논리(predicate logic)라고 함

예제 2-24 'x는 3보다 크다' 는 술어임을 보이자.

풀이 변수 x에 1을 배정하면 진리값이 거짓인 명제 '1은 3보다 크다' 가 되며, 변수 x가 4로 치환되면 진리값이 참인 명제 '4는 3보다 크다' 가 된다. 따라서 술어이다.

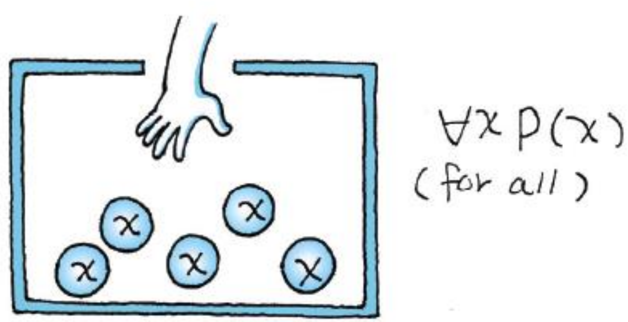
예제 2-25 x가 실수라 가정하고, '$x^2+1>0$' 을 $p(x)$라고 했을 때 $p(x)$의 진리값이 거짓인 x의 값을 구해보자.

풀이 $x^2+1>0$은 x가 실수일 경우 모든 x에 대하여 참이므로, $p(x)$의 진리값은 항상 참이다. 그러므로 $p(x)$의 진리값이 거짓인 x는 없다.

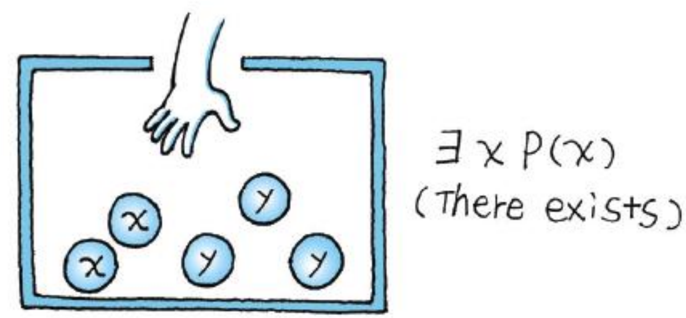
2.6 술어 논리

술어 한정자(Predicate Quantifier)

- 술어를 나타내는 방법 중에서 변수의 범위를 한정시키는 것임
- 한정자에는 '모든 것에 대하여(for all)' 와 '존재한다(there exist)' 의 두 가지가 있음
- '모든 것에 대하여' 는 기호 ∀를사용
- '존재한다' 는 기호 ∃로 나타냄



모든 x에 대하여 $P(x)$가 성립



어떤 x에 대하여 $P(x)$가 성립

정의 2-6 변수 x가 가질 수 있는 모든 값에 대하여 $p(x)$의 전체 한정자(universal quantifier)는 다음과 같이 나타내는 명제를 말한다.

'모든 x에 대하여 $p(x)$는 참이다'

전체 한정자의 기호는 '$\forall$'로 표시하고, $p(x)$에 대한 전체 한정자는 $\forall x\ p(x)$로 나타내며, $\forall x\ p(x)$가 참이 되기 위한 필요충분조건은 술어 $p(x)$가 x의 전체 집합 U에 대하여 성립하여야 한다.

예제 2-26 x가 정수라고 할 때, 다음 명제에 대해서 참과 거짓을 판별해보자.

(1) $\forall x[x < x + 1]$

(2) $\forall x[x = 3]$

(3) $\forall x[x > x - 3]$

풀이 (1) 모든 정수 x에 대해 $x+1$은 항상 x보다 크므로 참이다.

(2) x의 값이 3 이외의 경우라면 '$x=3$' 은 거짓이다. 즉, 모든 x에 대해 성립하는 것은 아니므로 거짓이다.

(3) 모든 x에 대하여 x는 $x-3$보다 크므로 참이다.

정의 2-7 변수 x가 가질 수 있는 모든 값에 대하여 $p(x)$의 존재 한정자(existential quantifier)는 다음과 같이 나타내는 명제를 말한다.

'어떤 x에 대하여 $p(x)$가 참인 x가 존재한다'

존재 한정자의 기호는 '∃'로 표시하고, $p(x)$에 대한 존재 한정자는 $\exists x\ p(x)$로 나타내며, $\exists x\ p(x)$가 참이기 위한 필요충분조건은 전체 집합 U 안에 $p(x)$를 만족시키는 x가 적어도 한 개 존재해야 한다.

여기서 잠깐!!

'모든' 또는 '임의의' 라는 뜻을 가진 'All' 을 의미하는 A를 뒤집어 놓은 것을 기호 ∀로 표현하며 전체 한정자라 하고, '존재한다' 라는 뜻을 가진 'Exist' 를 의미하는 영문자 E를 바꾸어 놓은 것을 기호 ∃로 표현하며 존재 한정자라고 한다.

예제 2-27

x가 정수이고 $p(x)$가 '$x = x^2$'이라고 할 때 다음 명제의 진리값을 구해보자

(1) $\forall x \ \ p(x)$

(2) $\exists x \ \ p(x)$

풀이 (1) 이 명제는 '모든 x에 대하여 $x = x^2$이다'. 그러나 $x = 2$일 때는 $x = x^2$이 성립하지 않으므로 이 명제는 거짓이다.

(2) 이 명제는 '$x = x^2$인 정수 x가 존재한다' 이다. $x = 0$일 때 $x = x^2$이 성립하므로 이 명제는 참이다.

예제 2-28 술어 한정자를 사용하여 다음과 같이 논리적 표기를 할 때, 그에 대한 명제를 서술해보자.

(1) $\exists x \ \ p(x, y)$

(2) $\sim(\forall x \ \ p(x))$

(3) $\exists y \ (\forall x \ \ p(x, y))$

풀이 (1) $p(x, y)$가 성립하는 x가 존재한다.

(2) 모든 x에 대하여 $p(x)$가 성립하는 것은 아니다.

(3) 모든 x에 대하여 $p(x, y)$가 성립하는 y가 존재한다.

정의 2-8 전체 한정자에서 '모든 x에 대하여 $p(x)$가 성립한다'고 하면, 그의 부정은 '$p(x)$가 성립하지 않는 x가 존재한다'가 된다. 이것을 논리 기호를 사용하여 나타내면 다음과 같다.

$\sim(\forall x \ \ p(x)) \Leftrightarrow \exists x \, (\sim p(x))$

정의 ②-9 존재 한정자에서 '$p(x)$가 성립하는 x가 존재한다'라고 하면, 그의 부정은 '모든 x는 $p(x)$가 성립하지 않는다'가 된다. 이것을 논리 기호를 사용하여 나타내면 다음과 같다.

$\sim(\exists x\ \ p(x)) \Leftrightarrow \forall x(\sim p(x))$

예제 ②-29 x는 '학생은'이고, $p(x)$는 'x는 공부한다'일 때 다음 문장의 부정을 서술하고, 그 부정을 논리적 기호로 표시해보자.

(1) 모든 학생은 공부한다.

(2) 공부를 하는 학생이 존재한다.

풀이 (1) 이 명제의 부정은 '공부하지 않는 학생도 있다'이다. 이것을 논리적 기호로 표시하면, $\sim(\forall x\ \ p(x)) \Leftrightarrow \exists x(\sim p(x))$이 된다.

(2) 이 명제의 부정은 '모든 학생은 공부를 하지 않는다'가 된다. 논리적 기호로 표시하면, $\sim(\exists x\ \ p(x)) \Leftrightarrow \forall x(\sim p(x))$이 된다.

2.7 논리용 언어 - Prolog

- 프롤로그(Prolog)

  논리와 명제를 컴퓨터 프로그램을 통해 보다 빠르고 쉽게 구현할 수 있는 프로그래밍 언어임

- Prolog의 특징
  - 사실(fact), 규칙(rule), 질문(question)들로 프로그램이 구성됨
  - 사실과 규칙들을 데이터베이스로 구성하였으며, 프로그램 실행은 자료에 대한 질문의 응답 형식임
  - 대화식의 명령 방식을 사용함
  - 사용자의 질문에 답하기 위해 추론 엔진(inference engine)을 사용하고 사용자가 사실과 규칙 등을 입력함

2.7 논리용 언어 - Prolog

- Prolog 프로그램
  - 우리가 쉽게 접하기 드문 색다른 형태의 언어임
  - 논리적 연산과 인공지능 분야에서의 논리 판단을 위한 프로그램의 구현에 있어서 매우 중요한 역할을 담당함
  - 사실과 규칙들만 기술하고 프로그램의 실행 순서는 명시하지 않으므로 '제 5세대 컴퓨터 언어' 라고도 함

〈표 2.11〉 Prolog 프로그램의 구현 예

human(socrates).	– 소크라테스는 사람이다.(사실)
human(daesukim).	– 김대수는 사람이다.(사실)
animal(wurry).	– 워리는 동물이다.(사실)
animal(X):–human(X).	– 모든 사람은 동물이다.(추론 규칙, X는 변수)
die(X):–animal(X).	– 모든 동물은 죽는다.(추론 규칙)
?–die(socrates).	– 소크라테스는 죽는가를 질의함
yes	– 시스템의 대답: yes
?–die(X).	– 모든 죽는 것의 이름 X를 질의함
X=socrates.	
X=daesukim.	
X=wurry.	

요약

- 명제란 어떤 사고를 나타내는 문장 중에서 참이나 거짓을 객관적이고 명확히 구분할 수 있는 문장이나 수학적 식을 말한다.
- 하나의 문장이나 식으로 구성되어 있는 명제를 단순 명제라 하고, 여러 개의 단순 명제들이 논리 연산자들로 연결되어 만들어진 명제를 합성 명제라고 한다.
- 명제 p가 주어졌을 때 그 명제에 대한 부정은 명제 p의 반대되는 진리값을 가지며, 기호로는 $\sim p$라 쓰고 'p가 아니다'라고 읽는다.
- 두 명제 p, q가 '그리고(AND)'로 연결되어 있을 때 명제 p, q의 논리곱은 $p \wedge q$로 표시하고, 'p 그리고 q'라고 읽는다.
- 두 명제 p, q가 '또는(OR)'으로 연결되어 있을 때 명제 p, q의 논리합은 $p \vee q$로 표시하고, 'p 또는 q'라고 읽는다.

요약

- 두 명제 p, q의 배타적 논리합은 $p \oplus q$로 표시하고 '익스클루시브 OR(exclusive OR) 또는 XOR' 이라고 읽는다.
- 조건 연산자를 함축이라고도 한다. 임의의 명제 p, q의 조건 연산자는 $p \rightarrow q$로 표시하고, 'p이면 q이다' 라고 읽는다.
- 명제 p, q의 쌍방 조건은 $p \leftrightarrow q$로 표시하고, 'p이면 q이고, q이면 p이다' 라고 읽는다.
- 명제 $p \rightarrow q$에 대하여 $q \rightarrow p$를 역, $\sim p \rightarrow \sim q$를 이, $\sim q \rightarrow \sim p$를 대우라 한다. 어떤 명제와 그 명제의 대우는 동치 관계이다.
- 합성 명제에서 그 명제를 구성하는 단순 명제들의 진리값에 관계없이 그 합성 명제의 진리값이 항상 참의 값을 가질 때 그 명제를 항진 명제라 하고, 진리값이 항상 거짓의 값을 가질 때 그 명제를 모순 명제라고 한다.

요약

- 두 개의 명제 p, q의 쌍방 조건 $p \leftrightarrow q$가 항진 명제이면, 두 명제 p, q는 논리적 동치라 하고, $p \equiv q$ 또는 $p \Leftrightarrow q$라고 표시한다. 즉, 명제 p와 q는 같은 논리값을 가진다.
- 주어진 명제가 참인 것을 기반으로 하여 새로운 명제가 참이 되는 것을 유도해 내는 방법을 추론이라 하며, $p \rightarrow q$, $q \vdash p$와 같이 나타낸다.
- 주어진 전제가 참이고 결론도 참인 추론을 유효 추론이라 하고 추론이 거짓이면 허위 추론이라고 한다.
- 추론의 주요 법칙으로는 긍정 법칙, 부정 법칙, 삼단 법칙 등이 있다.
- 명제를 p라 할 때 $p(x)$를 변수 x에 대한 명제 술어라고 하는데, 명제 논리와 구분하여 명제 술어에 대한 논리를 술어 논리라고 한다.

요약

- 변수 x가 가질 수 있는 모든 값에 대하여 $p(x)$의 전체 한정자는 '모든 x에 대하여 $p(x)$는 참이다' 와 같이 나타내는 명제를 말한다. $p(x)$에 대한 전체 한정자는 $\forall x \ p(x)$로 나타낸다.
- 변수 x가 가질 수 있는 모든 값에 대하여 $p(x)$의 존재 한정자는 '어떤 x에 대하여 $p(x)$가 참인 x가 존재한다' 와 같이 나타내는 명제를 말한다. $p(x)$에 대한 존재 한정자는 $\exists x \ p(x)$로 나타낸다.
- 프롤로그는 논리와 명제를 컴퓨터 프로그램을 통해 보다 빠르고 쉽게 구현할 수 있는 프로그래밍 언어로서, 논리적 판단을 위한 프로그램의 구현에 있어서 매우 중요한 역할을 담당한다.

응용

- 논리 분야 응용들
  - 알고리즘의 설계나 증명
  - 디지털 논리 회로 설계
  - 논리 프로그램 관련 분야
  - 관계형 데이터베이스 이론
  - 오토마타와 계산 이론
  - 인공지능 등 다양한 분야에 필요한 이론적 기반을 제공함