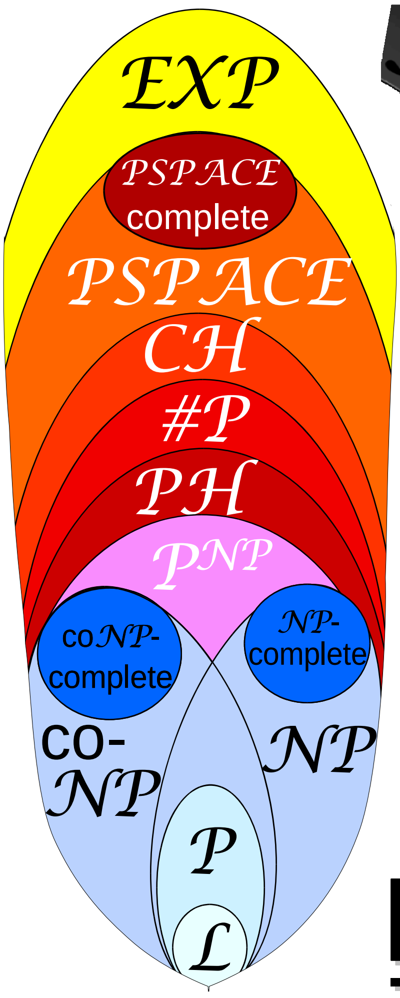
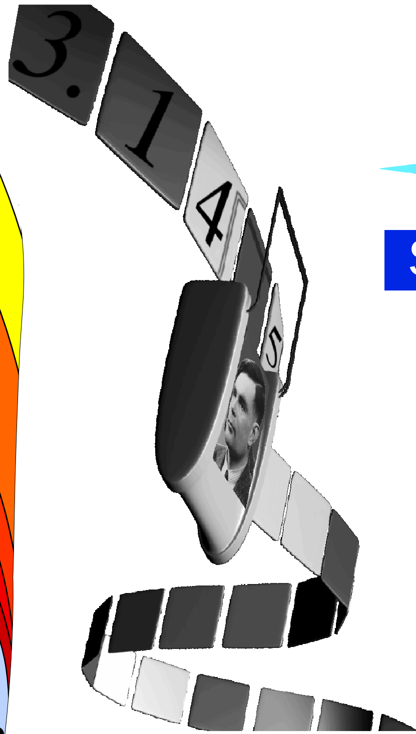




KAIST

School of **Computing**

Martin Ziegler
박세원
신승우
조준희
(박찬수)

Complexity and Real Computation

Please ask questions!

KAIST
LOGIC: M.Ziegler

Informal Logic

a) cDonald: The sum of two even primes is a square

b) 2 mothers and 2 daughters together buy 3 hats, yet each receives her own.

c) Smart phones are prohibited during exams: A students' phones are collected initially — and only half of them returned. Still, nobody complains!

d) All pink unicorns can fly!

Tautology
Satisfiability
Inconsistency

e) Epimenides the Cretan said: *All Cretans are liars*!

f) Can you correctly answer this very question?

Kobayashi-Maru

g) Is "*no*" the only correct answer to this question?

Boolean Logic

KAIST
LOGIC: M.Ziegler

- Truth values 0 and 1
- operations like $\vee$, $\wedge$, $\neg$

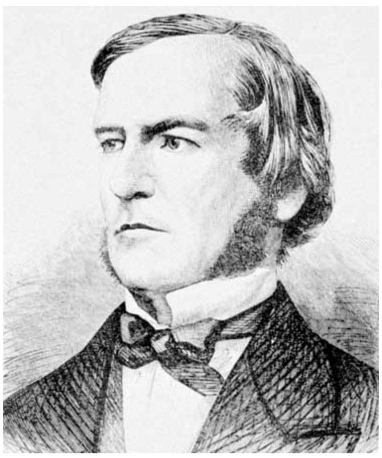
Example (expression):
$(\neg(0\vee 1) \wedge (1\wedge\neg 0)) \vee \neg 1$

x	$\neg x$
0	1
1	0

x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

$$((x \wedge y) \vee (\neg x\wedge\neg y)) \wedge ((x \wedge z) \vee (\neg x\wedge\neg z))$$



x	y	z	$\neg x\wedge\neg y$	$(x\wedge y)\vee(\neg x\wedge\neg y)$	$(x \wedge z)\vee(\neg x\wedge\neg z)$
0	0	0	1	1	1
0	0	1	1	1	0
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	0	1
1	1	0	0	1	0
1	1	1	0	1	1

Propositional "Programming"

- $\bigvee_{1\leq k\leq n} x_k := x_1 \vee x_2 \vee \ldots \vee x_n$ "at least one"
- $\bigwedge_{1\leq k\neq \ell\leq n} \neg(x_k \wedge x_\ell)$ "at most one"
- $x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge x_4$ "$x_1=1$ & $x_2=0$ & $x_3=0$ & $x_4=1$"
- $h_{b_1,\ldots,b_n}(x_1,\ldots,x_n)$ "$x_1=b_1$ & $x_2=b_2$ & … & $x_n=b_n$"
- $\bigvee_{\underline{b}:f(\underline{b})=1} h_{\underline{b}}(\underline{x}) = f(\underline{x})$

Can express every Boolean function using only $\vee$, $\wedge$, $\neg$

"$x \Rightarrow y$" = "$y \vee \neg x$"

"$x \Leftrightarrow y$" = "$(x \wedge y) \vee (\neg x \wedge \neg y)$"

Satisfiability and Tautology

A propositional formula $\varphi(x_1,\ldots,x_n)$ is satisfiable if there exists an assignment of its variables $x_1,\ldots,x_n$ (with **0**s and **1**s) that makes φ evaluate to **true**.

A set Φ of formulae is satisfiable if there exists a (joint) assignment to all occurring variables that makes every $\varphi\in\Phi$ evaluate to **true**.

A tautology φ evaluates to **true** for every assignment.

Examples: a) $x\vee y$ satisfiable, no tautology
b) $x\vee\neg x\vee\neg y$ satisfiable and tautology
c) $x\wedge\neg x$ neither satisfiable nor tautology
d) $\{ x\vee y, x\vee\neg y, \neg x\vee y, \neg x\vee\neg y \}$ not satisfiable

Observe: φ is not tautology iff $\neg\varphi$ is satisfiable

Sequent Calculus

Formalize "*expressions implies other expression(s)*":

Def: Let Φ,Ψ denote sets of propositional formulae φ,ψ.

Write "$\Phi \models \Psi$" if every assignment of variables making all $\varphi\in\Phi$ **true**, also renders at least one $\psi\in\Psi$ **true**.

Examples: a) $\{x\wedge y\}\models\{x\}$ b) $\{\varphi\wedge\psi\}\models\{\varphi\}$

c) $\{\varphi\vee\psi\}\models\{\varphi,\psi\}$ d) $\{x\vee y, x\vee\neg y\}\models\{x\}$

e) $\{\}\models\psi$ iff ψ tautology f) $\Phi\models\{\}$ iff Φ not satisfiable

Observation (sound rules):

a) $\Phi,\varphi\models\Psi,\varphi$

b) $\Phi\models\Psi,\varphi \;\to\; \Phi,\neg\varphi\models\Psi$

c) $\Phi,\varphi\models\Psi \;\to\; \Phi\models\Psi,\neg\varphi$

d) $\Phi\models\Psi,\varphi,\psi \;\to\; \Phi\models\Psi,\varphi\vee\psi$

e) $\Phi,\varphi,\psi\models\Psi \;\to\; \Phi,\varphi\wedge\psi\models\Psi$

f) $\Phi,\varphi\models\Psi$ & $\Phi,\psi\models\Psi \;\to\; \Phi,\varphi\vee\psi\models\Psi$

g) $\Phi\models\Psi,\varphi$ & $\Phi\models\Psi,\psi \;\to\; \Phi\models\Psi,\varphi\wedge\psi$

Abbr Φ,φ for $\Phi\cup\{\varphi\}$

Examples of Formal Proofs

"**Theorem:**" $x\models(x\wedge y)\vee\neg y$ semantic proof: truth table

Syntactic proof by deduction/application of rules:

a) $x,y\models y$ c) $x\models y,\neg y$ a) $x\models x,\neg y$ g) $x\models x\wedge y,\neg y$ d) $x\models(x\wedge y)\vee\neg y$ ■

"**Theorem:**" $\psi\models\neg\neg\psi$ a) $\psi\models\psi$ b) $\psi,\neg\psi\models\{\}$ c) $\psi\models\neg\neg\psi$

"**Theorem:**" $\neg\neg\psi\models\psi$ a) $\psi\models\psi$ c) $\{\}\models\psi,\neg\psi$ b) $\neg\neg\psi\models\psi$

"**Theorem:**" $\varphi\vee\psi\models\psi\vee\varphi$

Sound Rules:

a) $\Phi,\varphi\models\Psi,\varphi$

b) $\Phi\models\Psi,\varphi \;\to\; \Phi,\neg\varphi\models\Psi$

c) $\Phi,\varphi\models\Psi \;\to\; \Phi\models\Psi,\neg\varphi$

d) $\Phi\models\Psi,\varphi,\psi \;\to\; \Phi\models\Psi,\varphi\vee\psi$

e) $\Phi,\varphi,\psi\models\Psi \;\to\; \Phi,\varphi\wedge\psi\models\Psi$

f) $\Phi,\varphi\models\Psi$ & $\Phi,\psi\models\Psi \;\to\; \Phi,\varphi\vee\psi\models\Psi$

g) $\Phi\models\Psi,\varphi$ & $\Phi\models\Psi,\psi \;\to\; \Phi\models\Psi,\varphi\wedge\psi$

Sound and Complete Proof System

"**Theorem:**" $x \models (x\wedge y)\vee\neg y$ vs. $\forall \mathbf{x,y}.\ x \Rightarrow (x\wedge y)\vee\neg y$

(Meta) Theorem (proven by *structural induction*)**:**
$\Phi\models\psi$ is true iff it can be derived from the rules.
Such a derivation can be found algorithmically!

Additional rules for *Predicate Logic* (Quantifiers):

$\Phi\models\Psi,\psi(\underline{x},\mathbf{0})\vee\psi(\underline{x},\mathbf{1}) \;\rightarrow\; \Phi\models\Psi,\exists y.\psi(\underline{x},y)$

$\Phi\models\Psi,\psi(\underline{x},\mathbf{0})\wedge\psi(\underline{x},\mathbf{1}) \;\rightarrow\; \Phi\models\Psi,\forall y.\psi(\underline{x},y)$

....

a) $\Phi,\varphi\models\Psi,\varphi$

b) $\Phi\models\Psi,\varphi \;\rightarrow\; \Phi,\neg\varphi\models\Psi$

c) $\Phi,\varphi\models\Psi \;\rightarrow\; \Phi\models\Psi,\neg\varphi$

d) $\Phi\models\Psi,\varphi,\psi \;\rightarrow\; \Phi\models\Psi,\varphi\vee\psi$

e) $\Phi,\varphi,\psi\models\Psi \;\rightarrow\; \Phi,\varphi\wedge\psi\models\Psi$

f) $\Phi,\varphi\models\Psi$ & $\Phi,\psi\models\Psi \;\rightarrow\; \Phi,\varphi\vee\psi\models\Psi$

g) $\Phi\models\Psi,\varphi$ & $\Phi\models\Psi,\psi \;\rightarrow\; \Phi\models\Psi,\varphi\wedge\psi$

First-Order Logic

So far just Boolean operations $\forall \mathbf{x,y}.\ x \Rightarrow (x\wedge y)\vee\neg y$
and variables ranging over Boolean values **0** and **1**.

Examples: a) $(\{\mathbf{0},\mathbf{1}\}, \mathbf{0}, \mathbf{1}, \vee, \wedge, \neg, =)$ Booleans

b) $(\mathbb{R}, 0, 1, +, -, \times, <)$ Reals as a ring

c) $(\mathbb{C}, 0, 1, +, -, \times, =)$ Complex numbers

d) $(\mathbb{N}, 0, 1, +, \times, <)$ Peano Arithmetic

e) $(\mathbb{N}, 0, 1, +, <)$ Presburger Arithmetic

f) $(\mathbb{R}^{2\times 2}, 0, I, +, \times, =)$ Real square matrices

g) $(\mathbb{Q}, <)$ Rationals as linearly ordered set

(Constants are functions of arity 0, $+$ and $\times$ of 2.)

(Meta)Definition: A **structure** is a set X with functions $f_k{:}X^{\sigma_k}\rightarrow X$ & relations $R_\ell \subseteq X^{\tau_\ell}$ of arities σ_k,τ_k

Data Structures <u>are</u> Structures

E.g. a **stack** S storing elements from D,

with methods **new**$\in S$, **push**$:S\times D\to S$, **pop**$:S\to D$

considered as functions on structure $X := S \cup D$.

Property/Axiom: **pop**(**push**$(s,d))=d$

pop(**new**) may return *any* element of D

which can be prevented: **if** $(s$!=**new**$)$ d:=**pop**(s)

Alternatively, consider method/relation **empty**$\subseteq S$.

(Meta)Definition: A **structure** is a set X with functions $f_k:X^{\sigma_k}\to X$ & relations $R_\ell \subseteq X^{\tau_\ell}$ of arities σ_k,τ_k

Expressions and Formulae

Examples: a) $\mathbb{B}=(\{\mathbf{0},\mathbf{1}\},\mathbf{0},\mathbf{1},\vee,\wedge,\neg,=)$

b) $(\mathbb{R},0,1,+,-,\times,<)$ $\qquad \forall x\,\exists y.\ x+y=0$

c) $(\mathbb{C},0,1,+,-,\times,=)$ $\qquad \forall x.\,(x=0 \ \vee\ \exists y.\ x\times y=1)$

d) $(\mathbb{N},0,1,+,\times,<)$ $\qquad \forall x\,\exists y.\ x=y\times y$

f) $(\mathbb{R}^{2\times 2},0,I,+,\times,=)$ $\qquad \forall x\,\forall y.\ x\times y=y\times x$

h) (S,**new**,**push**,**pop**,=) $\qquad \forall y\,\forall z.\,(x\neq y\times z \ \vee\ y=1\ \vee\ z=1)$

a) expr. x and $(x\wedge y)\vee\neg y$, formula $\forall x.\ x=(x\wedge y)\vee\neg y$

An **expression** is (syntactially valid) composed from functions; a **formula** is a Boolean/quantified combination of relations among expressions.

(Meta)Definition: A **structure** is a set X with functions $f_k:X^{\sigma_k}\to X$ & relations $R_\ell \subseteq X^{\tau_\ell}$ of arities σ_k,τ_k

Axiomatizing/Abstract Data Types

Examples: a) $\mathbb{B}=(\{\mathbf{0},\mathbf{1}\},\mathbf{0},\mathbf{1},\vee,\wedge,\neg,=)$

b) $(\mathbb{R},0,1,+,-,\times,<)$

c) $(\mathbb{C},0,1,+,-,\times,=)$

d) $(\mathbb{N},0,1,+,\times,<)$

f) $(\mathbb{R}^{2\times 2},0,I,+,\times,=)$

h) $(S$,**new**,**push**,**pop**,=)

$$\forall x\,\exists y.\ x+y=0$$
$$\forall x.\,(\,x=0\ \vee\ \exists y.\ x\times y=1\,)$$
$$\forall x\,\exists y.\ x=y\times y$$
$$\forall x\,\forall y.\ x\times y=y\times x$$
$$\forall y\,\forall z.\,(\,x\neq y\times z\ \vee\ y=1\ \vee\ z=1\,)$$

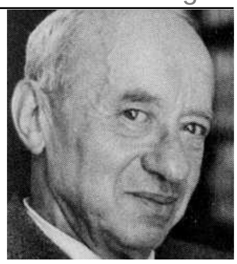
Hide implementation details: **0/1** vs. current on/off, **stack**=array/single/double linked list,… ℕ un/bin-ary

User must <u>only</u> rely on **axioms** of **abstract** data type: sufficient to *capture its properties up to isomorphism*.

(Meta)Definition: A **structure** is a set X with functions $f_k:X^{\sigma_k}\to X$ & relations $R_\ell\subseteq X^{\tau_\ell}$ of arities σ_k,τ_k

Hilbert, Gödel, Tarski

$(\mathbb{N},0,1,+,\times,<)$, $(\mathbb{R},0,1,+,\times,<)$



Theorem (*Tarski-Seidenberg*): There is an infinite decidable family Φ of axioms, such that the below claim holds for ℝ.

The theory of real numbers is decidable!

Theorem: There exists <u>no</u> (even *semi*-) decidable family Φ of axioms, such that the below claim holds for ℕ.

The theory of integers is <u>un</u>decidable!

Hilbert Program (1921): identify set Φ of axioms for **real** numbers such that:

$\mathbb{R}\models\psi$ iff ψ can be <u>derived</u> from Φ syntactically and such derivation can be found algorithmically.

Giuseppe Peano and Kurt Gödel

LOGIC: M.Ziegler

2nd Idea: Peano's Axioms for $(\mathbb{N}, 0, \underbrace{S}_{\text{successor/increment}}, +, \times)$

i) $\forall n\colon S(n)\neq 0$

ii) $\forall n,m\colon n=m \vee S(n)\neq S(m)$

iii) $\forall n\, \exists m\colon\ n=0 \vee n=S(m)$

iv) $\forall n\colon\ n=n+0$

v) $\forall n,m\colon n+S(m)=S(n+m)$

vi) $\forall n\colon\ n\times 0=0$

vii) $\forall n,m\colon n\times S(m) = n\times m + n$

$\mathbb{N} \models\ f(0)=0 \wedge \forall n.\ f(S(n))=f(n)+n\ \Rightarrow \forall n.\ 2\times f(n) = n\times S(n)$ **?**

First-order Logic: only quantification over elements, <u>not</u> over subsets/relations/functions!

~~**1st Idea:** Take as Φ *all* valid formulae!~~

Theorem: There exists <u>no</u> *semi*-decidable family Φ of axioms, such that it holds:
$\mathbb{B} \models \psi$ iff ψ can be <u>derived</u> from Φ syntactically.

Consequences and Conclusion

LOGIC: M.Ziegler

$(\mathbb{N}, 0, 1, +, \times, <)$

The ∃ theory of integers is <u>un</u>decidable!

$(\mathbb{R}, 0, 1, +, \times, <)$

(Algebr.) theory of real numbers is decidable!

$(\mathbb{Q}, 0, 1, +, \times, <)$

The ∃ ∀ ∃ theory of rationals is undecidable.

There are correct algorithms over $(\mathbb{N}, 0, 1, +, \times, <)$ whose total correctness however cannot be verified!

Correctness of loop invariants and ‚algorithms' over $(\mathbb{R}, 0, 1, +, \times, <)$ is decidable!

Other infinite structures? It depends … or is open!

Theoretical Computer Science for Numerics

- IEEE float/double: fast but awkward semantics, violates distributive law
- Interval arithmetic: error propagation
- Multiprecision arithmetic: how choose initial precision?

Programming language for *real* computation:
- imperative
- abstr. data type **REAL**
- computable semantics
- non-extensionality
- **formal verification**

- Stream computing/Recursive Analysis: No practical acceptance
- realRAM/BSS-Machine: *un*computable semantics

이계식, 김선영, 박세원...

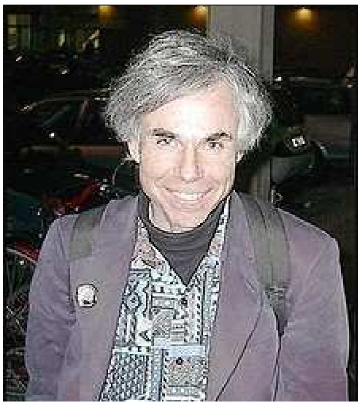
Folklore: Don't test for equality!

So, how about *in*equality "**<**" ?

$$x=0 \;\Leftrightarrow\; \neg(x<0) \;\wedge\; \neg(x>0)$$

$\rightarrow$ *non*-extensional semantics

Thanks for your attention!

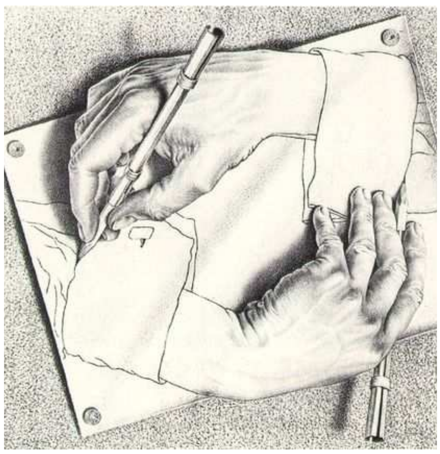


The Strange Loop phenomenon occurs whenever, by moving upwards (or downwards) through levels of some hierarchial system, we unexpectedly find ourselves right back where we started.

(Douglas Hofstadter)

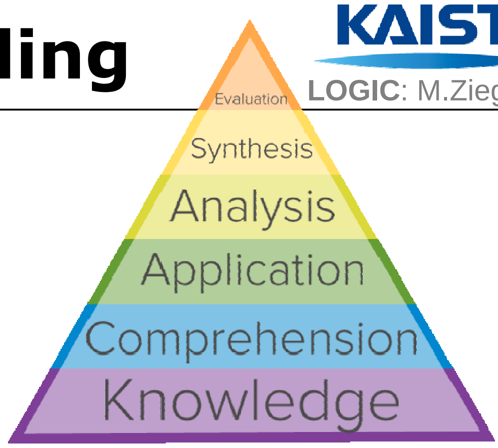
LOGIC

- **Logic**
- Set Theory
- Analysis
- Geometry
- (linear) Algebra
- Combinatorics
- Stochastics
- Number Theory
- **Theory of Computing**



Levels of Understanding

1. reproduce
2. apply
3. transfer
4. extend



Bloom's Taxonomy

- *What is thought is not said*
- *What is said is not heard*
- *What is heard is not understood*
- *What is understood is not believed*
- *What is believed is not yet advocated*
- *What is advocated is not yet acted on*
- *What is acted on is not yet completed*

Konrad Lorenz

(Nobel Prize 1973)