Course Introduction
Introduction to ML for CS

Dr. 何明昕, He Mingxin, Max

CS104: program07 @ yeah.net

CS108: mxhe1 @ yeah.net

Email Subject：CS104/108 + *ID* + *Name*: *TOPIC*

Sakai: CS104/CS108-数理逻辑导论/(H)

Introduction to Mathematical Logic (I2ML)

Exercise 01: Warm-Up Reading

- textA_ch01-BasicLogicConcepts.pdf (in 1 week)
- reading_BriefHistoryOfLogic-Vardi-2003.pdf (in 1 week)
- textF_Appendix-SetTheory.pdf
- textH_ch0-UsefullFactsAboutSets.pdf

} (at least one in 2~3 weeks if needed)

- reading+UnusualEffectivenessOfLogicInComputerScience-Halpern..Vardi..-2001.pdf (in 4-5 weeks)

Course Structure

- **Lectures**: 1-16周
  - CS104，星期五上午　3-4节，　一教110
  - CS108，星期四晚上　9-10节，三教205

Lecturer: **何明昕**, HE Mingxin, Max

CS104: program07 @ yeah.net

CS108: mxhe1 @ yeah.net

Teaching Tutor: **沈昀** (SHEN Yun) 老师

sheny @ mail.sustech.edu.cn

Course Information

Course Site in **Sakai**:

CS104-数理逻辑导论

CS108-数理逻辑导论(H)

Please Make sure the Proper Site is in your Working Space on sakai, or ask the instructor for help!

2 Course WeChat Groups for CS104 or CS108 Respectively. Join YOUR Course Group PLS!

Pre-requisites: Basics of Naive Set Theory. (If no idea on Sets, Learn it by yourself in early weeks PLS.)

Course Objective :

Logic, as the 1st fundamental discipline in the fields of science and technology, is the most important and indispensable theoretical foundation for many disciplines, including philosophy, math, CS & IS.

The I2ML course gives students an elementary introduction to the historical background, the basic concepts and principles, the major methodology, and important theoretical results of classical mathematical logic(CML).

It provides a good foundation for students to study modern logic and its application in depth, and also provides students with basic qualities of using logic methodology for further study and research of theoretical subjects in various disciplines and also help to establish a better theoretical foundation on logic and math for students (even with less math foundation) who are interested in engineering and technical domains.

Course Objective :

The teaching objectives of I2ML are:

(1) Let students know the historical background, essence/ nature, purpose, basic assumptions, scope, and major methodology of logic.

(2) Let students know the essence, purpose, basic assumptions, scope, and application fields of classical ML (CML).

(3) Let students be familiar with the propositional calculus in CML.

(4) Let students be familiar with the first-order predicate calculus (FOPC) in CML with some basic applications in computer science.

(5) Let students know the limitations of CML.

Expected Outputs :

The learning outcomes of the I2ML course:

(1) When encounter any problem, students can identify those logic-related elements in the problem by applying related the logic knowledge and can avoid or minimize logical mistakes.

(2) Students can use CML to formally represent knowledge in empirical fields and construct formal theories for the empirical fields.

(3) Students can use CML (and automatic reasoning/proof tools, if possible) to solve reasoning/proof problems in empirical fields.

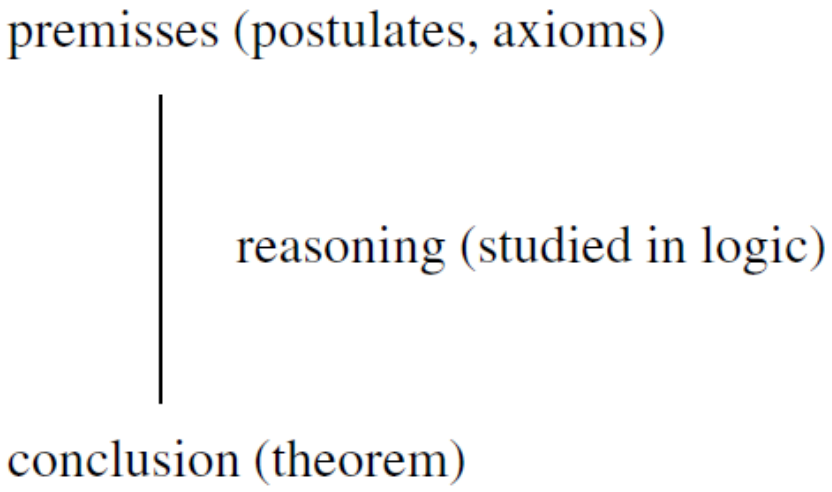
(4) Students can clearly identify those difficult issues in empirical field applications that are due to the limitations of CML.

(5) Students can further study various branches of modern logic and their applications based on knowledge and skills acquired in I2ML.

What is Logic? (from ChatGPT & textC)

what is logic?

Logic is the study of reasoning and argumentation. It is the process of using reason and critical thinking to analyze ideas, find connections, and draw conclusions. It is used in many fields and disciplines, such as philosophy, mathematics, and computer science.

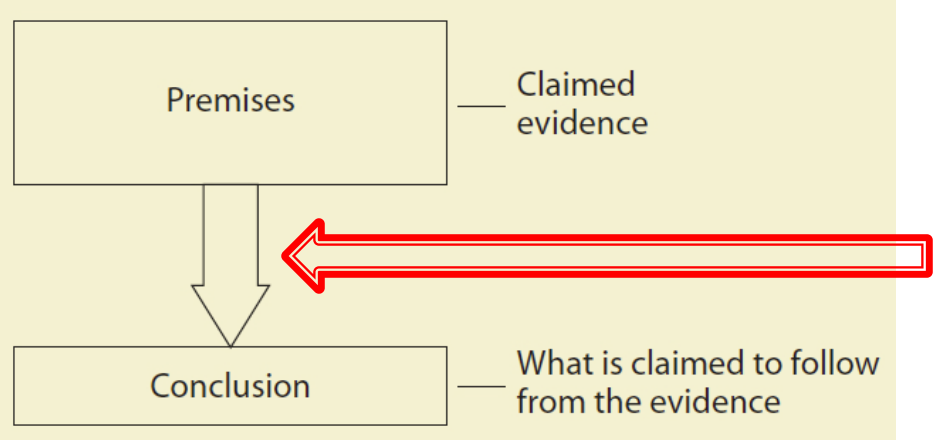


The word "logic" comes from the Greek word *Logykos, which* means "pertaining to reasoning."

The study of logic is the study of reasoning.

The basic question is what conclusions can be drawn with absolute certainty from a particular set of premisses.

What is Logic?



[From P. J. Hurley,
"A Concise Introduction to Logic"]

What entails what?

What follows from what?

Why? What are the evaluation criteria?

How to establish/define the evaluation criteria?

How to evaluate arguments/reasoning?

It is LOGIC to answer these fundamental questions.

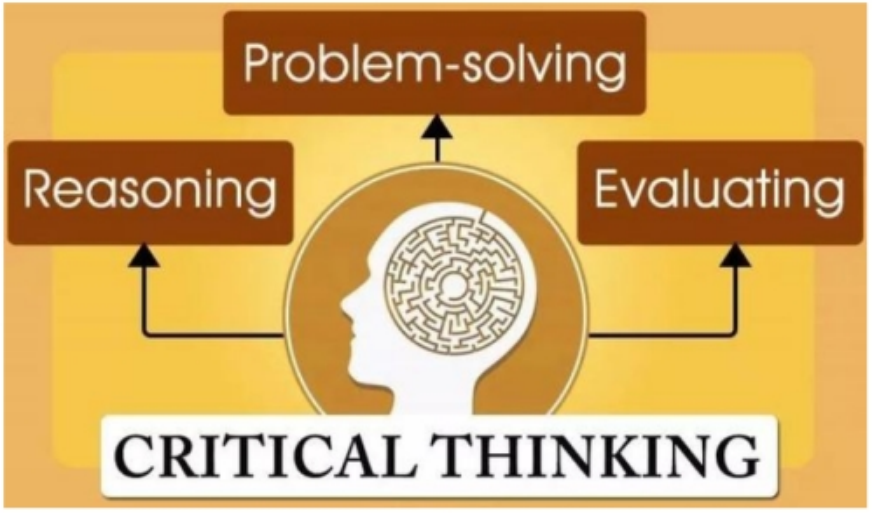
Study of Reasoning

Reasoning is the process of **using facts, evidence, and logic to form conclusions, solve problems, and make decisions**. It involves identifying better solutions, reasoning through potential outcomes, and using sound judgment.

Inference, Proof,
Discovery, Predication,
Argument, Deduction,
Induction, Abduction,
Fallacies, Fair Discussion..

推理是运用事实、证据和逻辑来形成结论、解决问题、作出决定的过程。它涉及确定更好的解决方案，通过潜在的结果进行推理，并使用正确的判断。

- Distinguish **Study, Learn, Train** ...
- One of the key element of Critical Thinking



- 批判性思维
  关键性思考
  建设性思考

- Critical Observing:
  建设性观察

- 系统化认知问题 与
  构建解决方案的能力

Critical thinking is a mental process involving the evaluation of arguments or statements to determine whether they are valid, accurate, or logical.

It involves the ability to think analytically, identify and assess arguments, draw conclusions, and make decisions.

Critical thinking is essential in many areas, such as problem-solving, decision-making, creativity, and research.

Logic, Symbolic Logic, Mathematical Logic

- Have you ever said to someone, “be logical"?
  - whatever your intuition was that is logic
- Mathematization/Formalization of the intuition is **mathematical logic (ML)** aka Symbolic Logic.

Logic from two perspectives

Logic from two perspectives: a practitioner and a logician

A practitioner cares about use of logic:

- Use logic to model specific things
- Determining if two formulas are equivalent or in a specific Normal Form (NF, DNF, CNF, Horn Formulas)
- Deduct a conclusion from a set of premises

A logician cares about properties of logic:

- Does every well-formed formula have a unique construction?
- Can this set of connectives construct any formula?
- Is every formula I can prove true? Can I prove every true formula? (Soundness & Completeness; Satisfiable? Decidability?)

Traditional logic

- "**Logic**" originates from ancient greek $\lambda o \gamma \iota \kappa \eta$ (logike).
- the study of the principles of reasoning; a part of philosophy; **philosophical logic**.
- explicit analyses of the principles of reasoning was developed in three cultures: **China**, **India**, and **Greece**.
- modern treatment descends from the Greek tradition, particularly **Plato-Aristotelian logic**, which was further developed by **Islamic logicians** and then **medieval European logicians**.
- mostly studied with rhetoric (修辞学), the syllogism (三段论) and philosophy.

Mathematical Logic

Mathematical logic began to diverge as a distinct field in the mid-19th century, motivated by the interest to the **foundations of mathematics**.

- a subfield of logic and mathematics.
- has numerous applications in mathematics, computer science, linguistics and philosophy.
- Unifying themes include:
  - the expressive power of formal logics
  - the deductive power of formal proof systems

- Antient Greece (around 400 BC):
  - Aristotle's theory of syllogisms
  - Euclid's axioms for planar geometry
- 1700s:
  Leibniz and Lambert treated logic in an algebraic way
- 19th century:
  - G. Boole and A. De Morgan presented systematic mathematical treatments of propositional logic
  - G. Frege and B. Russell developed logic with quantifiers

- late 19th century:
  axiomatic frameworks for fundamental areas of mathematics such as **geometry**, **arithmetic** and **analysis** were developed.
  - Arithmetic: Peano (1888)
  - Geometry: Lobachevsky (1826); Hilbert (1899)
  - Analysis: Weierstrass, Bolzano, Cauchy, Dedekind

  Also Georg Cantor developed the fundamental concepts of infinite **set theory**.

- early 20th century:
  - paradoxes in informal set theory: Russell paradox – the 3rd crisis of mathematics.
  - Hilbert's program: prove the consistency of foundational theories.
  - Hilbert's 23 problems: 1, 2, 10, 17
  - Axiomatic set theory: Zermelo-Fraenkel set theory (ZF)
  - K. Gödel, G. Gentzen etc: provided partial resolution to Hilbert's program, and clarified the issues involved in proving consistency
  - Turing machine: birth of modern computer

Great Logicians

David Hilbert

1862 - 1943

Ernst F. F. Zermelo

1871 - 1953

Bertrand A.W. Russell

1872 - 1970

Alfred Tarski

1902 - 1983

Kurt Gödel

1906 - 1978

Alan Turing

1912 - 1954

Subfields

Contemporary mathematical logic is roughly divided into four areas:

- **set theory**: sets
- **model theory**: models of various theories
- **recursion theory**: the properties of (in)computable functions and the Turing degrees
- **proof theory**: formal proofs in various logical deduction systems

Different focus, but no sharp border line. These areas share basic results on logic, particularly first-order logic, and definability.

Why should CS students study logic?

- Logic is fun!
- Logic improves one's ability to think analytically and to communicate precisely.
- Logic has many applications in Computer Science.

Why should CS students study logic?

Differential equations
are the calculus of
Newtonian physics

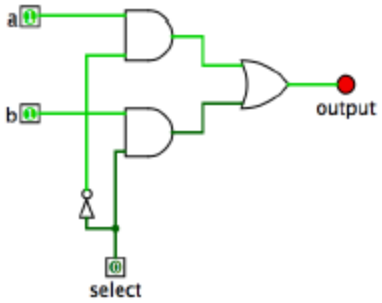
Logic
is the calculus of
computer science

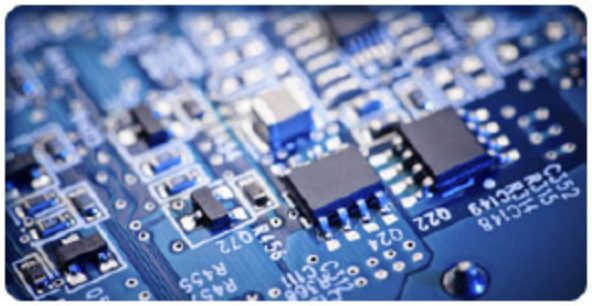
Logic provides tools to define/manipulate computational objects.

Logic and computer science

Circuit Design

- Digital circuits are the basic building blocks of an electronic computer.





- Computer Organization and Design
  Operating Systems

Logic and computer science

Databases

- Structural Query Language (SQL) $\approx$ first-order logic
- Efficient query evaluation based on relational algebra
- Scale to large databases with parallel processors

- Introduction to Database Management
  Database Systems Implementation

Logic and computer science

Type Theory in Programming Language

- Propositions in logic $\leftrightarrow$ types in a programming language
- Proofs of a proposition $\leftrightarrow$ programs with the type
- Simplifications of proofs $\leftrightarrow$ evaluations of the programs

- The compiler course
  Principles of Programming Languages
  Compiler Construction

Logic and computer science

Artificial Intelligence

- Logic as a fundemental form to knowledge Representation
- Automated reasoning and Theorm Proving
- ChatGPT from OpenAI is the hottest AI product now in weeks.
- IBM Watson won the Jeopardy Man vs. Machine Challenge
- Artificial Intelligence
  Machine Learning

Logic and computer science

Formal verification

- Prove that a program is bug free. Bugs can be costly and dangerous in real life.
- Intel's Pentium FDIV bug (1994) cost them half a billion dollars.
- Cancer patients died due to severe overdoses of radiation.
- Theory of Computing (Finite Automata)

Logic and computer science

Algorithms and Theory of Computing

- How much time and memory space do we need to solve a problem?
- Are there problems that cannot be solved by algorithms?
- Algorithm Design and Analysis
  Introduction to the Theory of Computing

Propositional Logic (PL)

Below we give some concrete simple arguments from different fields.

a1) If 1 = 2, then I am the Pope of Rome.
I am not the Pope of Rome.
Therefore: not 1 = 2.

a2) If 1 = 2, then I am the Pope of Rome.
Not 1 = 2.
Therefore: I am not the Pope of Rome.

b1) If triangle ABC is equiangular, then it is isosceles.
Triangle ABC is not isosceles.
Therefore: Triangle ABC is not equiangular.

b2) If triangle ABC is equiangular, then it is isosceles.
Triangle ABC is not equiangular.
Therefore: Triangle ABC is not isosceles.

c1) If it snows, then it is cold.
It is not cold.
Therefore: It does not snow.

c2) If it snows, then it is cold.
It does not snow.
Therefore: It is not cold.

Arguments a1, b1 and c1 in above Example have the same (valid) reasoning structure:

Pattern of reasoning Called **Modus Tollens**

if P_1, then P_2
not P_2
therefore: not P_1

$$\frac{P_1 \rightarrow P_2 \qquad \neg P_2}{\neg P_1}$$

Using $\rightarrow$ for 'if ..., then ...' and $\neg$ for 'not'

Arguments a2, b2 and c2 in above Example also have the same (invalid) reasoning pattern:

if P_1, then P_2	$P_1 \rightarrow P_2$
not P_1	$\neg P_1$
therefore: not P_2	$\neg P_2$

Why invalid?

Example 1.2 (some valid patters of reasoning).

	if P_1, then P_2	$P_1 \to P_2$	
1.	not P_2	$\underline{\quad\quad \neg P_2}$	*Modus Tollens*
	therefore: not P_1	$\neg P_1$	
	if P_1, then P_2	$P_1 \to P_2$	
2.	P_1	$\underline{P_1 \quad\quad}$	*Modus Ponens (MP)*
	therefore: P_2	P_2	
	P_1 if and only if (iff) P_2	$P_1 \rightleftarrows P_2$	
3.	not P_1	$\underline{\neg P_1 \quad\quad}$	
	therefore: not P_2	$\neg P_2$	
	not (P_1 and P_2)	$\neg(P_1 \wedge P_2)$	
4.	P_1	$\underline{\quad P_1 \quad\quad}$	
	therefore: not P_2	$\neg P_2$	
	P_1 or P_2	$P_1 \vee P_2$	
5.	not P_2	$\underline{\quad\quad \neg P_2}$	
	therefore: P_1	P_1	

We have introduced above $\rightleftarrows$ for 'if and only if (iff)', $\wedge$ for 'and', $\vee$ for the inclusive 'or', i.e., $P_1 \vee P_2$ stands for 'P_1 or P_2 or both P_1 and P_2'. The reader should verify that all patterns in Example 1.2 are valid.

The following two patterns of reasoning are frequently used in practice, although they are invalid:

if P_1, then P_2	$P_1 \rightarrow P_2$
not P_1	$\neg P_1$
therefore: not P_2	$\neg P_2$

if P_1, then P_2	$P_1 \rightarrow P_2$
P_2	P_2
therefore: P_1	P_1

So, the following concrete arguments are not correct:

If it rains, then the street becomes wet.
It does not rain.
Therefore: The street does not become wet.

If it is raining, then the street becomes wet.
The street becomes wet.
Therefore: It is raining.

Propositional Logic (PL)

We have introduced a new language for representing patterns of reasoning, the alphabet of which consists of the symbols:

$P_1, P_2, P_3, \ldots$	called atomic formulas
$\rightleftarrows, \rightarrow, \wedge, \vee, \neg$	called connectives
$(,)$	called parentheses.

REMARK: May have multiple different symbols to represent same connectives in different books:

$\rightleftarrows$, $\leftrightarrow$, $\equiv$, $\Leftrightarrow$	
$\rightarrow$, $\supset$, $\Rightarrow$	
$\wedge$,	$\bullet$
$\vee$,	$+$
$\neg$,	$\sim$

Formulas:

1. $P_1, P_2, P_3, \ldots$ are formulas. In other words, if P is an atomic formula, then P is a formula.
2. If A and B are formulas, then $(A \rightleftarrows B)$, $(A \to B)$, $(A \wedge B)$ and $(A \vee B)$ are formulas.
3. If A is a formula, then $(\neg A)$ is a formula too.

Example 1.3. P_1, P_3 and P_5 are formulas.

$(\neg P_1)$ and $(P_3 \to P_5)$ are formulas.

$((\neg P_1) \vee (P_3 \to P_5))$ is a formula.

Tautology, Contradiction, Satisfiable?

Predicate Logic (First-Order Logic, FOL)

All men are mortal.
Socrates is a man.
Therefore, Socrates is mortal.

The structure of the argument above is the following pattern:

For all objects x, if x is a person, then x is mortal.
Socrates is a person.
Therefore: Socrates is mortal.

$$\frac{\forall x[P(x) \rightarrow M(x)] \qquad P(c)}{M(c)}$$

Using $\forall x$ for 'for all x', $P(x)$ for 'x has the property P (to be a Person)', $M(x)$ for 'x has the property M (to be Mortal)' and c for 'Socrates'.

Next consider the following elementary argument:

John is ill
Therefore: someone is ill.

we need one more symbol: $\exists x$, for 'there is at least one x such that ...'.

$$\frac{I(c)}{\exists x[I(x)]}$$

Predicate Logic

SYMBOLS	NAME	MEANING
$x, \ldots$	individual variables	individuals in a given domain
$P, M, I, \ldots$	predicate symbols	predicates over the given domain
$c, \ldots$	individual constants	concrete individuals in the given domain
$\rightleftarrows; \rightarrow; \wedge; \vee, \neg$	connectives	iff; if ..., then ...; and; or; not
$\forall, \exists$	quantifiers	for all; there exists
$[,], (,)$	parentheses	

It turns out that one can select a few elementary steps of reasoning, among which

$\dfrac{A \quad A \rightarrow B}{B}$ called Modus Ponens, $\dfrac{A \wedge B}{B}$, $\dfrac{A}{A \vee B}$, $\dfrac{\forall x[A(x)]}{A(t)}$,

such that every valid pattern of reasoning, no matter how complex, can be built up from these elementary steps. This is Gödel's *Completeness Theorem*, 1930.

For instance, the following correct argument can be built up from the elementary steps just specified.

> John loves Jane and John is getting married.
> If John is getting married, then he is looking for another job.
> Hence: John is looking for another job or he does not love Jane.

The underlying pattern of reasoning is:

$$\frac{\begin{array}{l} P_1 \wedge P_2 \\ P_2 \rightarrow P_3 \end{array}}{P_3 \vee \neg P_1}$$

And indeed, this pattern can be built up from the elementary steps specified above as follows:

$$\dfrac{\dfrac{\dfrac{\text{premiss}}{P_1 \wedge P_2}}{P_2} \qquad \dfrac{\text{premiss}}{P_2 \rightarrow P_3}}{P_3}}{P_3 \vee \neg P_1}$$

And the four elementary steps of reasoning specified above can be supplemented by a few more elementary steps to form what is called Gentzen's [8] system of *Natural Deduction*

Example 1.23 Finally, we return to the argument of Examples 1.1 and 1.2, which can be coded up by the sequent $p \wedge \neg q \rightarrow r,\ \neg r,\ p \vdash q$ whose validity we now prove:

1	$p \wedge \neg q \rightarrow r$	premise
2	$\neg r$	premise
3	p	premise
4	$\neg q$	assumption
5	$p \wedge \neg q$	$\wedge$i 3, 4
6	r	$\rightarrow$e 1, 5
7	$\bot$	$\neg$e 6, 2
8	$\neg\neg q$	$\neg$i 4–7
9	q	$\neg\neg$e 8

(From page 23 of textB.)

The basic rules of natural deduction:

	introduction	*elimination*
$\wedge$	$\dfrac{\phi \quad \psi}{\phi \wedge \psi}\,\wedge\mathrm{i}$	$\dfrac{\phi \wedge \psi}{\phi}\,\wedge\mathrm{e}_1 \qquad \dfrac{\phi \wedge \psi}{\psi}\,\wedge\mathrm{e}_2$
$\vee$	$\dfrac{\phi}{\phi \vee \psi}\,\vee\mathrm{i}_1 \qquad \dfrac{\psi}{\phi \vee \psi}\,\vee\mathrm{i}_2$	$\dfrac{\phi \vee \psi \qquad \boxed{\begin{array}{c}\phi\\ \vdots\\ \chi\end{array}} \qquad \boxed{\begin{array}{c}\psi\\ \vdots\\ \chi\end{array}}}{\chi}\,\vee\mathrm{e}$
$\rightarrow$	$\dfrac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \psi\end{array}}}{\phi \rightarrow \psi}\,\rightarrow\mathrm{i}$	$\dfrac{\phi \quad \phi \rightarrow \psi}{\psi}\,\rightarrow\mathrm{e}$

The basic rules of natural deduction:

	introduction	*elimination*
$\neg$	$\dfrac{\boxed{\begin{array}{c}\phi\\ \vdots\\ \bot\end{array}}}{\neg\phi}\ \neg\mathrm{i}$	$\dfrac{\phi \quad \neg\phi}{\bot}\ \neg\mathrm{e}$
$\bot$	(no introduction rule for $\bot$)	$\dfrac{\bot}{\phi}\ \bot\mathrm{e}$
$\neg\neg$		$\dfrac{\neg\neg\phi}{\phi}\ \neg\neg\mathrm{e}$

Some useful derived rules:

$$\frac{\phi \to \psi \quad \neg\psi}{\neg\phi}\ \text{MT} \qquad \frac{\phi}{\neg\neg\phi}\ \neg\neg\text{i}$$

$$\frac{\begin{array}{|c} \neg\phi \\ \vdots \\ \bot \end{array}}{\phi}\ \text{PBC} \qquad \frac{}{\phi \vee \neg\phi}\ \text{LEM}$$

Figure 1.2. Natural deduction rules for propositional logic.

Another example: the argument above about Socrates which has as its underlying pattern of reasoning

$$\frac{\begin{array}{l}\forall x[P(x) \to M(x)] \\ P(c)\end{array}}{M(c)}$$

can be built up from the elementary steps in the system of Natural Deduction as follows:

$$\dfrac{\text{premiss}\quad P(c) \qquad \dfrac{\text{premiss}\quad \forall x[P(x) \to M(x)]}{P(c) \to M(c)}}{M(c)}$$

The schema above is called a logical *deduction* (in the system of Natural Deduction) of $M(c)$ from the premisses $\forall x[P(x) \to M(x)]$ and $P(c)$. We say that $M(c)$ is logically *deducible from* $\forall x[P(x) \to M(x)]$ and $P(c)$, since such a logical deduction exists.

Formal Language

Examples:

1. (Digital sequence understood by computer)

   $$00101010100000101111101000$$

2. (Programme Language, eg. Java or C)

   ```
   s = 1; i = n; while (i > 0) { s *= a; i--; }
   ```

3. (Propositional Logic)

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

4. (First-Order Logic)

   $$\forall\epsilon\exists\delta\forall x(|x-a| < \delta \rightarrow |f(x)-c| < \epsilon)$$

5. (Modal Logic)

   $$\neg(\Diamond p) \leftrightarrow \Box(\neg p)$$

Programming in Logic: Prolog

It was only in the early 1970's that the idea emerged to use the formal language of logic as a programming language. An example is PROLOG, which stands for PROgramming in LOGic. A logic program is simply a set of formulas (of a particular form) in the language of predicate logic. The formulas below constitute a logic program for kinship relations. The objects are people and there are two binary predicates 'parent of' (p), and 'grandparent of' (g).

- A_1: p(art, bob).
- A_2: p(art, bud).
- A_3: p(bob, cap).
- A_4: p(bud, coe).
- A_5: $g(x,z)$:- $p(x,y)$, $p(y,z)$.

'art', 'bob', 'bud', 'cap' and 'coe' are individual constants and A_5 stands for $p(x,y) \wedge p(y,z) \rightarrow g(x,z)$.

Programming in Logic: Prolog

Now if we ask the question

?- g(art, cap)

the answer will be 'yes', corresponding with the fact that g(art, cap) can be logically deduced from the premisses or data $A_1, \ldots, A_5$.

But if we ask the question

?- g(art, amy)

the answer will be 'no', corresponding with the fact that g(art, amy) cannot be logically deduced from $A_1, \ldots, A_5$. Note that this does not mean that $\neg g$(art, amy) logically follows from $A_1, \ldots, A_5$.

Program verification

$$\frac{(\!|\phi|\!)\ C_1\ (\!|\eta|\!) \qquad (\!|\eta|\!)\ C_2\ (\!|\psi|\!)}{(\!|\phi|\!)\ \ C_1; C_2\ \ (\!|\psi|\!)}\ \text{Composition}$$

$$\frac{}{(\!|\psi[E/x]|\!)\ x = E\ (\!|\psi|\!)}\ \text{Assignment}$$

$$\frac{(\!|\phi \wedge B|\!)\ C_1\ (\!|\psi|\!) \qquad (\!|\phi \wedge \neg B|\!)\ C_2\ (\!|\psi|\!)}{(\!|\phi|\!)\ \texttt{if}\ B\ \{C_1\}\ \texttt{else}\ \{C_2\}\ (\!|\psi|\!)}\ \text{If-statement}$$

$$\frac{(\!|\psi \wedge B|\!)\ C\ (\!|\psi|\!)}{(\!|\psi|\!)\ \texttt{while}\ B\ \{C\}\ (\!|\psi \wedge \neg B|\!)}\ \text{Partial-while}$$

$$\frac{\vdash_{\mathrm{AR}} \phi' \to \phi \qquad (\!|\phi|\!)\ C\ (\!|\psi|\!) \qquad \vdash_{\mathrm{AR}} \psi \to \psi'}{(\!|\phi'|\!)\ C\ (\!|\psi'|\!)}\ \text{Implied}$$

Figure 4.1. Proof rules for partial correctness of Hoare triples.

Main Topics Covered (Negotiable, to be tuned)

Basics of Logic

Propositional Logic

- Syntax of PL and Rules of Inference (philosopher's view)
- Natural Deduction & Semantics for PL (philo's & math's)
- Formal proofs (mathematician's view)
- Proof systems for PL and their properties (computer scientist's view)
- PL solvers aka SAT solvers (hacker's view, optional)

First-order Logic

- definition of FOL and syntactic properties (philosopher's view)
- proof systems for FOL, etc (computer scientist's view)
- first-order theorem provers (hacker's view)

Main Topics Covered (cont. Negotiable)

Limitation of CML: Godel's Incompleteness Theorems
Science and Hypothesis (use of logic)
Fallacies and Unfair Discussion Methods (use of logic)
Logic Programming (application in CS, optional)
Program Verification (application in CS)
Modal Logic (optional, CS108 only)

TextBooks

textA_IntroductionToLogic-14ed--Copi-Cohen-2011.pdf

textB_LogicInComputerScience--2ed-Huth-Ryan-2004.pdf

textC_PhilosophicalAndMathematicalLogic--Swart-2018.pdf

textD_LogicAndProof-R3.18.4--Avigad-Lewis-Doorn-2021.pdf

textE_MathematicalLogic-Preprint--Slaman-Woodin-2019.pdf

textF_MathematicalLogicForComputerScience-3ed--Ben-Ari-2012.pdf

textG_LogicForComputerScientists-Reprint--Schoning-2008.pdf

textH_AMathematicalIntroductionToLogic-2ed--Enderton-2001.pdf

Besides logic, I2ML will try to help

- Thinking and communicating precisely
- Problem solving
- Creative thinking
- Critical thinking

BUT this is only one side (teaching) of the course.
Another side (studying) is your participation and efforts on the course...

Course Evaluation:

- Class participation/Quizes/Exercises 25%
- Assignments 25% (on sakai course site with exact due date-time)
- Final exam 50%

Important !!!

- Most ideas will be covered in class but some details might be skipped. These details are covered in the necessary reading.
- Assignments, Quizes and Exercises should be done individually although mini-group discussions are allowed for Assignments and Exercises
- Any dishonest behaviour and cheating in assignments will be dealt with severely
- If you get an idea for a solution from others or online you must acknowledge the source in your submission

Regulations on Academic Misconduct in courses for Undergraduate Students.docx

Regulations on Academic Misconduct in courses for Undergraduate Students in the SUSTech Department of Computer Science and Engineering

From Spring 2022, the plagiarism policy applied by the Computer Science and Engineering department is the following:

*** If an undergraduate assignment is found to be plagiarized, the first time the score of the assignment will be 0.**

*** The second time the score of the course will be 0.**

*** If a student does not sign the Assignment Declaration Form or cheats in the course, including regular assignments, midterms, final exams, etc., in addition to the grade penalty, the student will not be allowed to enroll in the two CS majors through 1+3, and cannot receive any recommendation for postgraduate admission exam exemption and all other academic awards.**

*** If an undergraduate assignment is found to be plagiarized, the first time the score of the assignment will be 0.**

*** The second time the score of the course will be 0.**

*** If a student does not sign the Assignment Declaration Form or cheats in the course, including regular assignments, midterms, final exams, etc., in addition to the grade penalty, the student will not be allowed to enroll in the two CS majors through 1+3, and cannot receive any recommendation for postgraduate admission exam exemption and all other academic awards.**

What is OK, and what isn't OK?

It's OK to work on an assignment with a friend, and think together about the program structure, share ideas and even the global logic. At the time of actually writing the code, you should write it alone.

It's OK to use in an assignment a piece of code found on the web, as long as you indicate in a comment where it was found and don't claim it as your own work.

It's OK to help friends debug their programs (you'll probably learn a lot yourself by doing so).

DO NOT Share Your Workable Code to Friends

It's OK to show your code to friends to explain the logic, as long as the friends write their code on their own later.

It's NOT OK to take the code of a friend, make a few cosmetic changes (comments, some variable names) and pass it as your own work.

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

计算机科学与工程系

Department of Computer Science and Engineering

Undergraduate Students Declaration Form

This is_____ (student ID: _____, who has enrolled in _____ course of the Department of Computer Science and Engineering. I have read and understood the regulations on courses according to "Regulations on Academic Misconduct in courses for Undergraduate Students in the SUSTech Department of Computer Science and Engineering". I promise that I will follow these regulations during the study of this course.

Signature:

Date:

Exercise 01: Warm-Up Reading

- textA_ch01-BasicLogicConcepts.pdf (in 1 week)
- reading_BriefHistoryOfLogic-Vardi-2003.pdf (in 1 week)
- textF_Appendix-SetTheory.pdf
- textH_ch0-UsefullFactsAboutSets.pdf

} (at least one in 2~3 weeks if needed)

- reading+UnusualEffectivenessOfLogicInComputerScience-Halpern..Vardi..-2001.pdf (in 4-5 weeks)

Summary of Lecture 1

- Course Introduction
- Introduction to ML for CS Students

To be discussed in Lecture 2:

- Basics of Logic, Definition of PL, Rules of Inference