COSC 221 - Introduction to Discrete Structures

Lecture - Logic-04

Readings

- Propositional Logic: Sections 3.1, 3.2, 3.3
- Predicate Logic: Sections 3.4
- Computer Science Connections
  1. Computational Complexity (Section 3.3)
  2. Modern Compilers (Section 3.3)
  3. Game Trees (Section 3.4)

- Argument Form
- Inference Rules

Not covered in detail in the textbook, but important!

- Argument Form
- Inference Rules

Not covered in detail in the textbook, but important!

	If $3 > 1$, then $\sqrt{3} > 1$
	$3 > 1$
$\therefore$	$\sqrt{3} > 1$
	If 2 is even, then 3 is even
	2 is even
$\therefore$	3 is even

- Argument Form
- Inference Rules

	If $3 > 1$, then $\sqrt{3} > 1$
	$3 > 1$
$\therefore$	$\sqrt{3} > 1$
	If 2 is even, then 3 is even
	2 is even
$\therefore$	3 is even

Same pattern, different contexts

	$p \Rightarrow q$
	p
$\therefore$	q

- Argument Form
- Inference Rules

	If $3 > 1$, then $\sqrt{3} > 1$
	$3 > 1$
$\therefore$	$\sqrt{3} > 1$
	If 2 is even, then 3 is even
	2 is even
$\therefore$	3 is even

Same pattern, different contexts

	$p \Rightarrow q$
	p
$\therefore$	q

conclusion

- Argument Form
- Inference Rules

Same pattern, different contexts

	$p \Rightarrow q$
	p
$\therefore$	q

premises (or, hypotheses)

conclusion

	If $3 > 1$, then $\sqrt{3} > 1$
	$3 > 1$
$\therefore$	$\sqrt{3} > 1$
	If 2 is even, then 3 is even
	2 is even
$\therefore$	3 is even

- Argument Form
- Inference Rules

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \hline \therefore & q \end{array}$$

premises (or, hypotheses)

conclusion

Argument Form: $\varphi_1, \varphi_2, \cdots, \varphi_{n-1}, \varphi_n$

A sequence of propositions, containing one/more proposition variables.

- ▷ Argument Form
- ▷ Inference Rules

Same pattern, different contexts

premises (or, hypotheses)

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

conclusion

Argument Form: $\varphi_1, \ \varphi_2, \ \cdots, \ \varphi_{n-1}, \ \varphi_n$

A sequence of propositions, containing one/more proposition variables.

- Argument Form
- Inference Rules

Same pattern, different contexts

premises (or, hypotheses)

$$\begin{array}{l} p \Rightarrow q \\ p \\ \hline \therefore\ q \end{array}$$

conclusion

Valid Argument Form:

Whenever the premises are all true, the conclusion is true.

Argument Form: $\varphi_1, \varphi_2, \cdots, \varphi_{n-1}, \varphi_n$

A sequence of propositions, containing one/more proposition variables.

- Argument Form
- Inference Rules

Q.1) Is this argument form valid?

A) Yes

B) No

C) You haven't taught us yet!

D) Whatever!

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

Valid Argument Form:

Whenever the premises are all true, the conclusion is true.

Argument Form: $\varphi_1, \varphi_2, \cdots, \varphi_{n-1}, \varphi_n$

A sequence of propositions, containing one/more proposition variables.

- Argument Form
- Inference Rules

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

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

Valid Argument Form:

Whenever the premises are all true, the conclusion is true.

Testing Validity of Argument Forms

- Truth-Table
- Tautology

- ▷ Argument Form
- ▷ Inference Rules

Same pattern, different contexts

premises (or, hypotheses)

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \hline \therefore & q \end{array}$$

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

Testing Validity of Argument Forms

- ▷ Truth-Table
- ▷ Tautology

- ▷ Argument Form
- ▷ Inference Rules

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

Same pattern, different contexts

$$\begin{array}{l} p \Rightarrow q \\ p \\ \therefore q \end{array}$$

premises (or, hypotheses)

Conclusion

Testing Validity of Argument Forms

- ▷ Truth-Table
- ▷ Tautology

- ▷ Argument Form
- ▷ Inference Rules

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

premises (or, hypotheses)

Conclusion

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

Critical Row: premises are all true

Testing Validity of Argument Forms

- ▷ Truth-Table
- ▷ Tautology

- Argument Form
- Inference Rules

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

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

Critical Row: premises are all true

Testing Validity of Argument Forms

- Truth-Table
- Tautology

An argument form is valid iff the conclusion is true in every **critical row**

- Argument Form
- Inference Rules

Same pattern, different contexts

$$\begin{array}{ll} & p \Rightarrow q \\ & p \\ \therefore & q \end{array}$$

Testing Validity of Argument Forms

- Truth-Table
- Tautology

An argument form is valid iff it is a tautology when treated as a conditional proposition.

- Argument Form
- Inference Rules

For example,

$$(p \Rightarrow q) \wedge p \Rightarrow q$$

is indeed a tautology.

Same pattern, different contexts

$$\begin{array}{|lll|} & p & \Rightarrow q \\ & p & \\ \therefore & q & \\ \hline \end{array}$$

Testing Validity of Argument Forms

- Truth-Table
- Tautology

An argument form is valid iff it is a tautology when treated as a conditional proposition.

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

Modus Ponens (Latin meaning: method of affirming)

$p \Rightarrow q$	If $3 > 1$, then $\sqrt{3} > 1$
p	$3 > 1$
$\therefore \quad q$	$\therefore \quad \sqrt{3} > 1$

Propositional Logic

Inference Rules

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

Sound Argument

- Valid argument form
- Premises are all true

Modus Ponens

From Argument Form to Argument:

Substitute propositions for proposition variables

$p \Rightarrow q$	If $3 > 1$, then $\sqrt{3} > 1$
p	$3 > 1$
$\therefore \quad q$	$\therefore \quad \sqrt{3} > 1$

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

Sound Argument

- Valid argument form
- Premises are all true

Modus Tollens (Latin meaning: method of denying)

	$p \Rightarrow q$		If $\sqrt{.5} > 1$, then $.5 > 1$
	$\neg q$		$.5 \leq 1$
$\therefore$	$\neg p$	$\therefore$	$\sqrt{.5} \leq 1$

Propositional Logic

Inference Rules

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

Sound Argument

- Valid argument form
- Premises are all true

Elimination

$p \vee q$	$(x > 1) \vee (y > 1)$
$\neg q$	$x \leq 1$
$\therefore \quad p$	$\therefore \quad y > 1$

Propositional Logic

- Modus Ponens
- Modus Tollens
- Elimination
- Transitivity

Inference Rule: Valid Argument Form

Sound Argument

- Valid argument form
- Premises are all true

Transitivity

	$p \Rightarrow q$	If $x > 1$, then $\lvert x \rvert > 1$
	$q \Rightarrow r$	If $\lvert x \rvert > 1$, then $x \neq 0$
$\therefore$	$p \Rightarrow r$	$\therefore$ If $x > 1$, then $x \neq 0$

Generalization

	p	If $x > 1$, then $\lvert x \rvert > 1$
$\therefore$	$p \lor r$	$\therefore \quad (x > 1) \lor (x < 1)$

Specialization

	$p \land q$	$(x > 1) \land x$ even
$\therefore$	p	$\therefore \quad x > 1$

Proof by Division into Cases

	$p \vee q$	$(x \geq 1) \vee (x \leq -1)$
	$p \Rightarrow r$	If $x \geq 1$, then $x^2 \geq 1$
	$q \Rightarrow r$	If $x \leq -1$, then $x^2 \geq 1$
$\therefore$	r	$\therefore$ $x^2 \geq 1$

Exercise: Use the truth-table method to show the above is valid.

Contradiction Rule

	$\neg p \Rightarrow F$	
$\therefore$	p	

The Story: Finding my glasses?

RK = "I was reading in the kitchen"

RL = "I was reading in the living room"

GK = "My glasses are on the kitchen table"

GC = "My glasses are on the coffee table"

SB = "I saw them at breakfast"

The Story: Finding my glasses?

RK = “I was reading in the kitchen”

RL = “I was reading in the living room”

GK = “My glasses are on the kitchen table”

GC = “My glasses are on the coffee table”

SB = “I saw them at breakfast”

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

The Story: Finding my glasses?

RK = "I was reading in the kitchen"

RL = "I was reading in the living room"

GK = "My glasses are on the kitchen table"

GC = "My glasses are on the coffee table"

SB = "I saw them at breakfast"

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

Q.2) Inference rule used is

A) transitivity

B) modus tollens

C) elimination

D) modus ponens

Step 1

$$\begin{array}{lrcl} & RK & \Rightarrow & GK \\ & GK & \Rightarrow & SB \\ \therefore & RK & \Rightarrow & SB \end{array}$$

The Story: Finding my glasses?

RK = "I was reading in the kitchen"

RL = "I was reading in the living room"

GK = "My glasses are on the kitchen table"

GC = "My glasses are on the coffee table"

SB = "I saw them at breakfast"

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

Q.3) Inference rule used is

A) transitivity

B) modus tollens

C) elimination

D) modus ponens

Step 2

$$\begin{array}{ll} & RK \Rightarrow SB \\ & \neg SB \\ \therefore & \neg RK \end{array}$$

Propositional Logic

Use Inference Rules to Construct Complex Arguments

The Story: Finding my glasses?

RK = “I was reading in the kitchen”

RL = “I was reading in the living room”

GK = “My glasses are on the kitchen table”

GC = “My glasses are on the coffee table”

SB = “I saw them at breakfast”

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

Q.4) Inference rule used is

A) transitivity

B) modus tollens

C) elimination

D) modus ponens

Step 3

$$\begin{array}{ll} & \neg RK \\ & RL \vee RK \\ \therefore & RL \end{array}$$

The Story: Finding my glasses?

RK = "I was reading in the kitchen"

RL = "I was reading in the living room"

GK = "My glasses are on the kitchen table"

GC = "My glasses are on the coffee table"

SB = "I saw them at breakfast"

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

Q.5) Inference rule used is

A) transitivity

B) modus tollens

C) elimination

D) modus ponens

Step 4

$$\begin{array}{ll} & RL \Rightarrow GC \\ & RL \\ \therefore & GC \end{array}$$

The Story: Finding my glasses?

RK = "I was reading in the kitchen"

RL = "I was reading in the living room"

GK = "My glasses are on the kitchen table"

GC = "My glasses are on the coffee table"

SB = "I saw them at breakfast"

Known Facts

- $RK \Rightarrow GK$
- $GK \Rightarrow SB$
- $\neg SB$
- $RK \vee RL$
- $RL \Rightarrow GC$

Put Together: the argument

(1)	$RK \Rightarrow GK$	Given
(2)	$GK \Rightarrow SB$	Given
(3)	$RK \Rightarrow SB$	transitivity
(4)	$\neg SB$	Given
(5)	$\neg RK$	moduc tollens: (3), (4)
(6)	$RL \vee RK$	Given
(7)	RL	elimination: (5), (6)
(8)	$RL \Rightarrow GC$	Given
$\therefore$	GC	modus ponens:(7), (8)

Let x, y, z be three particular integers, satisfying

- $x < \ y \ < z$
- $z \leq 1$
- $|x| > 1$

To show that $x < -1$.

True Propositions

(a) $(x > 1) \ \Rightarrow \ (y > 1)$

(b) $(y > 1) \ \Rightarrow \ (z > 1)$

(c) $(z \leq 1)$

(d) $(x < -1) \ \vee (x > 1)$

Let x, y, z be three particular integers, satisfying

- $x < y < z$
- $z \leq 1$
- $|x| > 1$

To show that $x < -1$.

True Propositions

(a) $(x > 1) \Rightarrow (y > 1)$
(b) $(y > 1) \Rightarrow (z > 1)$
(c) $(z \leq 1)$
(d) $(x < -1) \vee (x > 1)$

The Argument

(1)	$(x > 1) \Rightarrow (y > 1)$	Given
(2)	$(y > 1) \Rightarrow (z > 1)$	Given
(3)	$(x > 1) \Rightarrow (z > 1)$	Transitivity
(4)	$\neg(z > 1)$	Given
(5)	$\neg(x > 1)$	Modus Tollens: (3),(4)
(6)	$(x < -1) \vee (x > 1)$	Given
$\therefore$	$(x < -1)$	Elimination: (5), (6)

Knights always tell truth and Knaves always tell lie. A says "B is a knight" B says "A and I are of opposite type".

What are A and B?

Knights always tell truth and Knaves always tell lie. A says "B is a knight"

B says "A and I are of opposite type".

What are A and B?

- Suppose A is a knight. B is a knight and "A and I are of opposite types" is true. A contradiction!

Knights always tell truth and Knaves always tell lie. A says "B is a knight"

B says "A and I are of opposite type".

What are A and B?

- Suppose A is a knight. B is a knight and "A and I are of opposite types" is true. A contradiction!
- $\therefore$ A is NOT a knight

Knights always tell truth and Knaves always tell lie. A says "B is a knight" B says "A and I are of opposite type".

What are A and B?

- Suppose A is a knight. B is a knight and "A and I are of opposite types" is true. A contradiction!
- $\therefore$ A is NOT a knight

Q.6) The inference rule used is

A) Modus Ponens
B) Elimination
C) Contradiction
D) Generalization

Knights always tell truth and Knaves always tell lie. A says "B is a knight"

B says "A and I are of opposite type".

What are A and B?

- Suppose A is a knight. B is a knight and "A and I are of opposite types" is true. A contradiction!
- $\therefore$ A is NOT a knight
- A is a knight or A is a knave

Knights always tell truth and Knaves always tell lie. *A* says "B is a knight"

B says "A and I are of opposite type".

What are A and B?

- Suppose *A* is a knight. *B* is a knight and "A and I are of opposite types" is true. A contradiction!
- $\therefore$ *A* is NOT a knight
- *A* is a knight or *A* is a knave
- $\therefore$ *A* is a knave

Knights always tell truth and Knaves always tell lie. A says "B is a knight" B says "A and I are of opposite type".

What are A and B?

- Suppose A is a knight. B is a knight and "A and I are of opposite types" is true. A contradiction!
- $\therefore$ A is NOT a knight
- A is a knight or A is a knave
- $\therefore$ A is a knave

Q.7) The inference rule used is

A) Modus Ponens
B) Elimination
C) Contradiction
D) Generalization

Converse Error	Inverse Error
$p \Rightarrow q$ q $\therefore\ p$	$p \Rightarrow q$ $\neg p$ $\therefore\ \neg q$
(affirming the consequence)	(denying the antecedent)

Concepts

- Propositions and Connectives
- Truth Values and Truth Tables
- Logic Equivalence and Laws

Arguments and Logic Inference

- Argument and Argument Form
- Inference Rules

Modelling and Inferences

- Wumpus-world, graph coloring
- iCliker exercises
- Reasoning and programming

- Statements $\Rightarrow$ Propositions
- Negation/Contrapositive of Compound (Conditional) Propositions
- Use of trueth tables and/or logic laws

Concepts

- ▷ Propositions and Connectives
- ▷ Truth Values and Truth Tables
- ▷ Logic Equivalence and Laws

Arguments and Logic Inference

- ▷ Argument and Argument Form
- ▷ Inference Rules

Modelling and Inferences

- ▷ Wumpus-world, graph coloring
- ▷ iCliker exercises
- ▷ Reasoning and programming

- ▷ Truth tables and Valid Argument Forms
- ▷ Construct logic arguments using inference rules

Additional Notes

Announcement 1: Quiz 1 (Monday, Feb 5th)

- Topics:
  - Propositional Logic
  - Basics of Predicate Logic

Logics and Automated Reasoning in AI

Knowledge Representation: KB (Knowledge Base)

Query and Question

Algorithmic Problem

Inference Engine

Logics and Automated Reasoning in AI

Knowledge Representation: KB (Knowledge Base)

Query and Question

Conjunction of facts and inference rules

Algorithmic Problem

Inference Engine

Logics and Automated Reasoning in AI

Knowledge Representation: KB (Knowledge Base)

Query and Question

Given a statement Q, is $KB \Rightarrow Q$ true?

Algorithmic Problem

Inference Engine

Logics and Automated Reasoning in AI

Knowledge Representation: KB (Knowledge Base)

Query and Question

Given a statement Q, is $KB \Rightarrow Q$ true?

Algorithmic Problem

Inference Engine

Is $KB \cap \neg Q$ False?

Why does this work?

Logics and Automated Reasoning in AI

Knowledge Representation: KB (Knowledge Base)

Query and Question

Given a statement Q, is $KB \Rightarrow Q$ true?

Algorithmic Problem

Inference Engine

A solver for the satisfiability problem.

E.g., $F = (x_1 \vee x_2) \wedge \ (x_8 \vee \neg x_{15} \vee x_{50}) \cdots \ (\neg x_{99} \vee x_{120})$

- **Imperative Programming**
- **Declarative Programming**

- **Imperative Programming**
- **Declarative Programming**

Java, C, C#, etc.
Instructions telling computers "how to do things"

- **Imperative Programming**
- **Declarative Programming**

Java, C, C#, etc.
Instructions telling computers "how to do things"

Prolog, LISP.
Statements telling computers "what to do"

- **Imperative Programming**
- **Declarative Programming**

Java, C, C#, etc.
Instructions telling computers "how to do things"

Prolog, LISP.
Statements telling computers "what to do"

Prolog - Programming in Logic

- Developed in 1970s
- Used in some areas of AI

Propositional Logic and Beyond

- **Imperative Programming**
- **Declarative Programming**

```
% A Mini Knowledgebase about Java Classes

%Rules
is-a(X, Y):- extends(X, Y).
is-a(X, Y):- extends(X, Z), is-a(Z, Y).
is-a(X, Y):- interface(Y), implements(X, Y).

%Facts
extends(mylist, arraylist).
extends(arraylist, abstractlist).
extends(error, throwable).
extends(exception, throwable).

implements(abstractlist, list).
implements(throwable, serializable).
interface(list).
```

Prolog - Programming in Logic

- Developed in 1970s
- Used in some areas of AI

- **Imperative Programming**
- **Declarative Programming**

```
% A Mini Knowledgebase about Java Classes

%Rules
is-a(X, Y):- extends(X, Y).
is-a(X, Y):- extends(X, Z), is-a(Z, Y).
is-a(X, Y):- interface(Y), implements(X, Y).

%Facts
extends(mylist, arraylist).
extends(arraylist, abstractlist).
extends(error, throwable).
extends(exception, throwable).

implements(abstractlist, list).
implements(throwable, serializable).
interface(list).
```

Propositions
E.g. "mylist extends arraylist"

Propositional Logic and Beyond

- **Imperative Programming**
- **Declarative Programming**

Propositions?

E.g. "X is a Y if X extends Y"

Remember "He is a student"

Propositions
E.g. "mylist extends arraylist"

```
% A Mini Knowledgebase about Java Classes

%Rules
is-a(X, Y):- extends(X, Y).
is-a(X, Y):- extends(X, Z), is-a(Z, Y).
is-a(X, Y):- interface(Y), implements(X, Y).

%Facts
extends(mylist, arraylist).
extends(arraylist, abstractlist).
extends(error, throwable).
extends(exception, throwable).

implements(abstractlist, list).
implements(throwable, serializable).
interface(list).
```

- **Imperative Programming**
- **Declarative Programming**

```
% A Mini Knowledgebase about Java Classes

%Rules
is-a(X, Y):- extends(X, Y).
is-a(X, Y):- extends(X, Z), is-a(Z, Y).
is-a(X, Y):- interface(Y), implements(X, Y).

%Facts
extends(mylist, arraylist).
extends(arraylist, abstractlist).
ble).
rowable).

st, list).
serializable).
```

Prolog Program

A collection of facts and rules

- **universal conditional statements**
- Next topic

Demo time! Go to SWISH

Prolog: Program Examples

```
% A Mini Knowledgebase about Java Classes

%Rules
is-a(X, Y):- extends(X, Y).
is-a(X, Y):- extends(X, Z), is-a(Z, Y).
is-a(X, Y):- interface(Y), implements(X, Y).

%Facts
extends(mylist, arraylist).
extends(arraylist, abstractlist).
extends(error, throwable).
extends(exception, throwable).

implements(abstractlist, list).
implements(throwable, serializable).
interface(list).
```

Prolog: Program Examples - Sudoku

```
%----------------------------------------------------------------
%Program Example: sudoku with CLP. It solves every
%soduko puzzle and to each puzzle, it can find all possible
%solutions. (Basic idea from the SWI-Prolog Manual)
%----------------------------------------------------------------
:- use_module(library(clpfd)). %import the CLP module

board([[_,_,3,_,_,_,_,_,_],
       [_,_,_,_,_,3,_,8,5],
       [_,_,1,_,2,_,_,_,_],
       [_,_,_,5,_,7,_,_,_],
       [_,_,4,_,_,_,1,_,_],
       [_,9,_,_,_,_,_,_,_],
       [5,_,_,_,_,_,_,7,3],
       [_,_,2,_,1,_,_,_,_],
       [_,_,_,_,4,_,_,_,6]]).
```

Sudoku Example Cont'd

```
%------------------------------------------------------------
% the predict to solve soduko
% -----------------------------------------------------------
solve:-
  board(Rows),
  length(Rows, 9), maplist(length_(9), Rows),
  append(Rows, Vs), Vs ins 1..9,

   sudoku(Rows),

   %solve the puzzle by labelling the variables
   label(Vs),

  %output the solution
  maplist(writeln, Rows).
```

References I