APPLICATIONS OF PROPOSITIONAL LOGIC

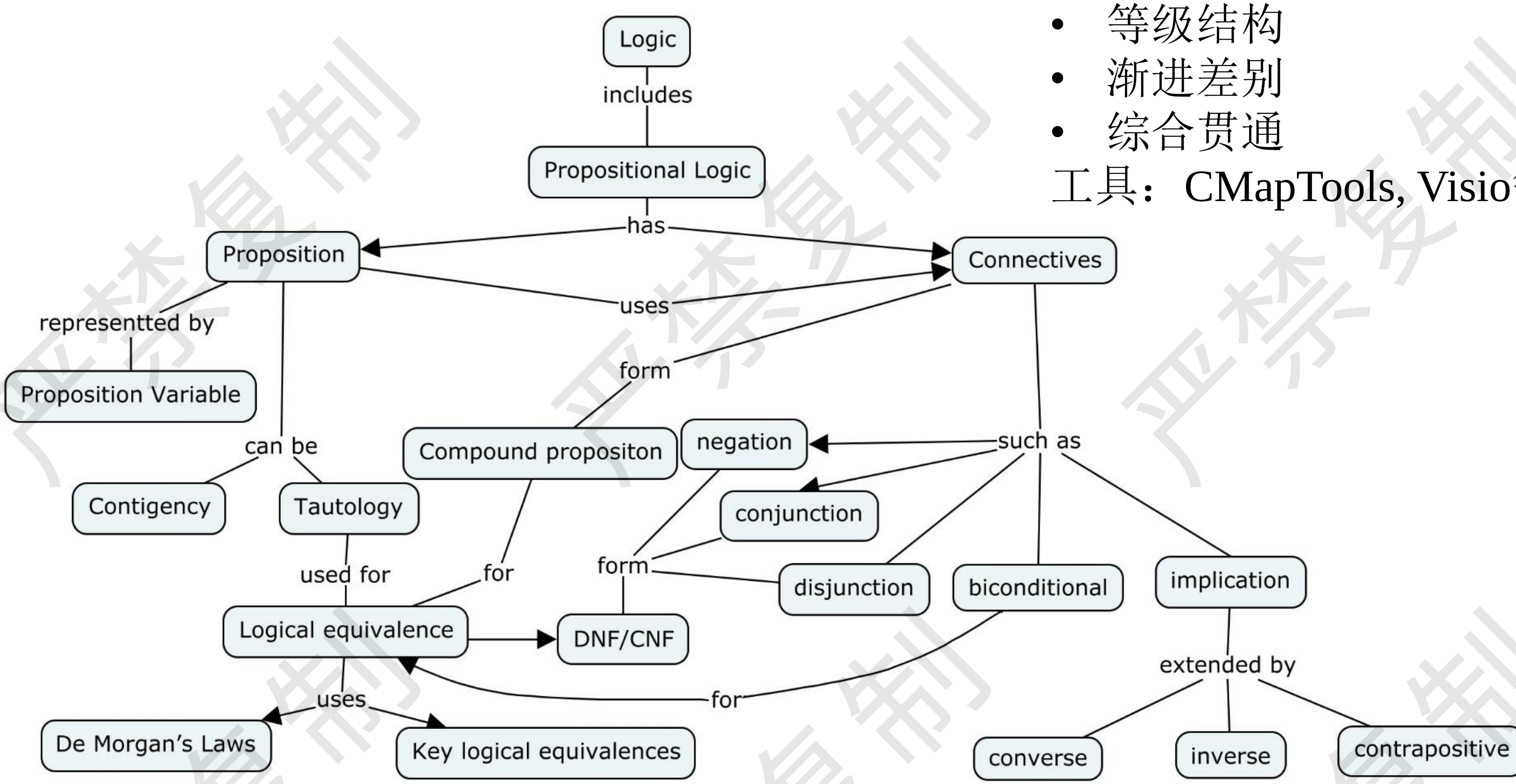
Section 1.3 (Chapter 1.2)

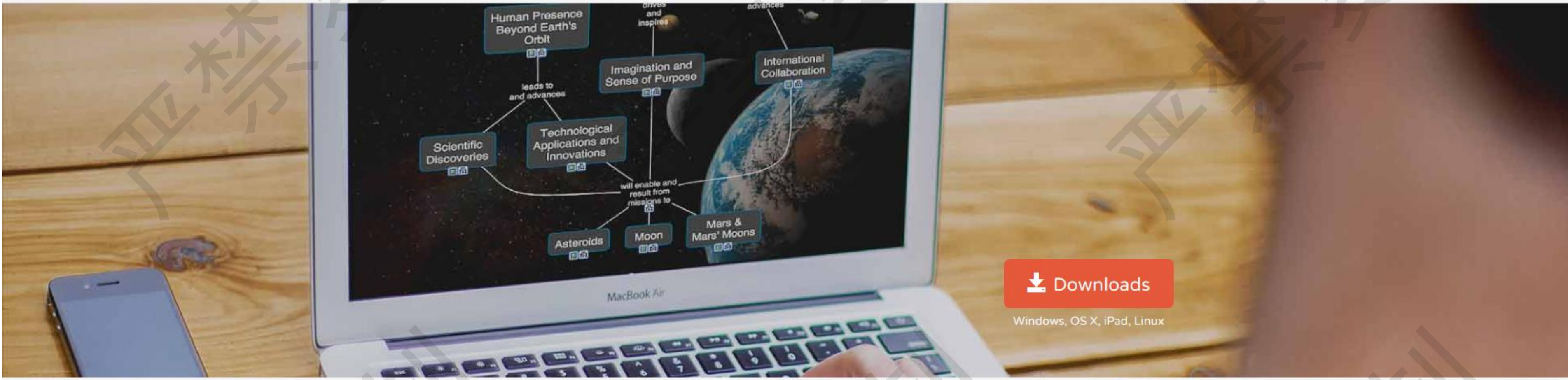
CONCEPT MAP FOR PROPOSITIONAL LOGIC

概念地图构图原则

- 等级结构
- 渐进差别
- 综合贯通

工具：CMapTools, Visio等





Downloads

Windows, OS X, iPad, Linux

Cmap software is a result of research conducted at the Florida Institute for Human & Machine Cognition (IHMC). It empowers users to construct, navigate, share and criticize knowledge models represented as concept maps.

URL: https://cmap.ihmc.us/

Cmap Products

Cmap products empowers users to construct, navigate, share and criticize knowledge models represented as concept maps. View our products to see how you can utilize our software in your work, studies, or research.

View Products

Learn About Concept Maps

Concept maps are graphical tools for organizing and representing knowledge in an organized fashion. Learn what concept maps are, how to construct them, and use them.

Learn More

Cmaps Around the World

Cmap software is used by individuals, schools, and institutions all around the world. See a variety a of uses of concept mapping and Cmap software by users of all ages.

Learn More

APPLICATIONS OF PROPOSITIONAL LOGIC: SUMMARY

- Translating English to Propositional Logic
- Boolean Search
- Logic Puzzles
- Logic Circuits

TRANSLATING ENGLISH SENTENCES

- Steps to convert an English sentence to a statement in propositional logic
  - Identify atomic propositions and represent using propositional variables.
  - Determine appropriate logical connectives
- "If I go to Harry's or to the country, I will not go shopping."

Solution

p: I go to Harry's

q: I go to the country.

r: I will go shopping.

If *p* or *q* then not *r*. $(p \vee q) \rightarrow \neg r$

MORE EXAMPLES

Write these system specifications in symbols using the propositions

- v: "The user enters a valid password,"
- a: "Access is granted to the user,"
- c: "The user has contacted the network administrator,"

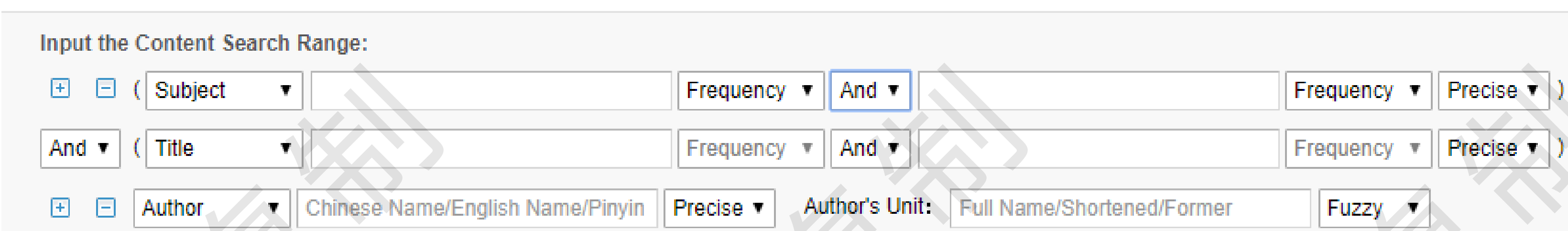
and logical connectives. Then determine if the system specifications are consistent.

(Definition: A list of propositions is consistent if it is possible to assign truth values to the proposition variables so that each proposition is true.)

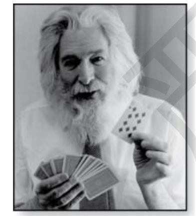
- (i) "The user has contacted the network administrator, but does not enter a valid password."
- (ii) "Access is granted whenever the user has contacted the network administrator or enters a valid password."
- (iii) "Access is denied if the user has not entered a valid password or has not contacted the network administrator."

BOOLEAN SEARCH

- Logical connectives are used extensively in searches of large collections of information
  - AND, OR, NOT
- Search engines
  - Baidu. AND: “ ”, OR: “ | ”, NOT: “ - ”
  - Google.
- Academic search
  - CNKI (www.cnki.net)
  - Web of Sciences (www.webofknowledge.com)



LOGIC PUZZLES



Raymond Smullyan (Born 1919)

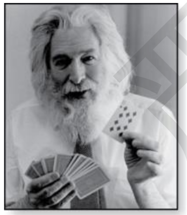
- An island has two kinds of inhabitants, *knights*, who always tell the truth, and *knaves*, who always lie.
- You go to the island and meet A and B.
  - A says "B is a knight."
  - B says "The two of us are of opposite types."

What are the types of A and B?

- If A tells you "We are not both truthtellers."

What are the types of A and B?

LOGIC PUZZLES



Raymond Smullyan (Born 1919)

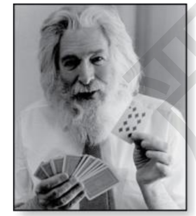
- You go to the island and meet A and B.
  - A says "B is a knight."
  - B says "The two of us are of opposite types."

What are the types of A and B?

Solution: Let p and q be the statements that A is a knight and B is a knight, respectively. So, then $\neg p$ represents the proposition that A is a knave and $\neg q$ that B is a knave.

- If A is a knight, then p is true. Since knights tell the truth, q must also be true. Then $(p \wedge \neg q) \vee (\neg p \wedge q)$ would have to be true, but it is not. So, A is not a knight and therefore $\neg p$ must be true.
- If A is a knave, then B must not be a knight since knaves always lie. So, then both $\neg p$ and $\neg q$ hold since both are knaves.

LOGIC PUZZLES



Raymond Smullyan (Born 1919)

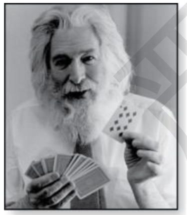
- You go to the island and meet A and B.
  - A says “B is a knight.”
  - B says “The two of us are of opposite types.”

What are the types of A and B?

Solution with propositional satisfiability:

- B is knight:
- The two of us are of opposite types
- A knight always tells true and a knave always tells false:
  - If A is a knight, then ‘B is a knight’ is T; If A is knave then “B is a knight” is F:
  - Similarly, for B:
- Final proposition:

LOGIC PUZZLES



Raymond Smullyan (Born 1919)

- If A tells you "We are not both truthtellers."

What are the types of A and B?

Solution: Let p and q be the statements that A is a knight and B is a knight, respectively. So, then $\neg p$ represents the proposition that A is a knave and $\neg q$ that B is a knave.

- We are not both truthtellers: $\neg(p \wedge q)$
- A knight always tells true and a knave always tells false: $\leftrightarrow$ (Equivalence)
- Final Proposition:

DISJUNCTIVE NORMAL FORM 析取范式

- Please fill in the compound proposition the meets the truth table.

p	*q*	?
T	T	**T**
T	F	**T**
F	T	**F**
F	F	**T**

DISJUNCTIVE NORMAL FORM 析取范式

- A propositional formula is in *disjunctive normal form* if it consists of a disjunction of (1, … ,*n*) disjuncts where each disjunct consists of a conjunction of (1, …, *m*) atomic formulas or the negation of an atomic formula.
  - $(p \wedge \neg q) \vee (p \wedge q)$

    Yes
  - $p \wedge (p \vee q)$

    No
- Disjunctive Normal Form is important for the circuit design methods.

DISJUNCTIVE NORMAL FORM

Example: Show that every compound proposition can be put in disjunctive normal form.

Solution: Construct the truth table for the proposition.

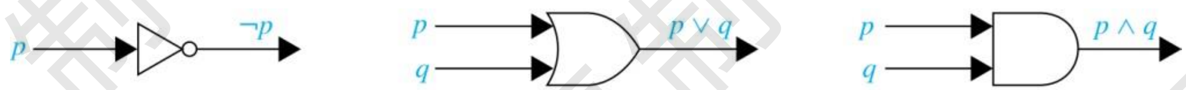
1. An equivalent proposition is the disjunction with ***n*** **disjuncts** (where *n* is the number of rows for which the formula evaluates to **T)**.
2. Each disjunct has *m* conjuncts where *m* is the number of distinct propositional variables.
3. Each conjunct includes the **positive form** of the propositional variable if the variable is assigned **T** in that row and the **negated form** if the variable is assigned **F** in that row.
4. This proposition is in disjunctive normal from.

CONJUNCTIVE NORMAL FORM 合取范式

- A compound proposition is in *Conjunctive Normal Form* (CNF) if it is a conjunction of disjunctions.
- Every proposition can be put in an equivalent CNF.
- Conjunctive Normal Form (CNF) can be obtained by eliminating implications, moving negation inwards and using the distributive and associative laws.
- Important in resolution theorem proving used in artificial Intelligence (AI).
- A compound proposition can be put in conjunctive normal form through repeated application of the logical equivalences covered earlier.

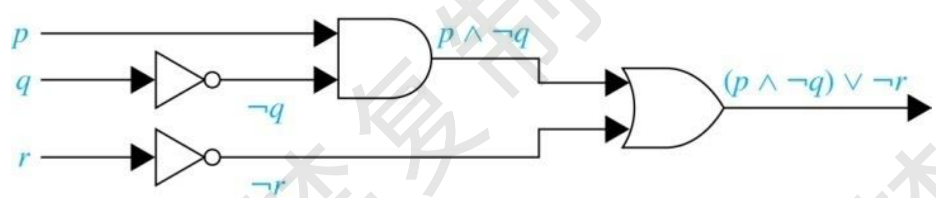
LOGIC CIRCUITS

- Electronic circuits; each input/output signal can be viewed as a 0 or 1.
  - 0 represents **False**
  - 1 represents **True**
- Complicated circuits are constructed from three basic circuits called gates.



Inverter   OR gate   AND gate

  - The inverter (**NOT gate**)takes an input bit and produces the negation of that bit.
  - The **OR gate** takes two input bits and produces the value equivalent to the disjunction of the two bits.
  - The **AND gate** takes two input bits and produces the value equivalent to the conjunction of the two bits.
- More complicated digital circuits can be constructed by combining these basic circuits to produce the desired output given the input signals by building a circuit for each piece of the output expression and then combining them. For example:

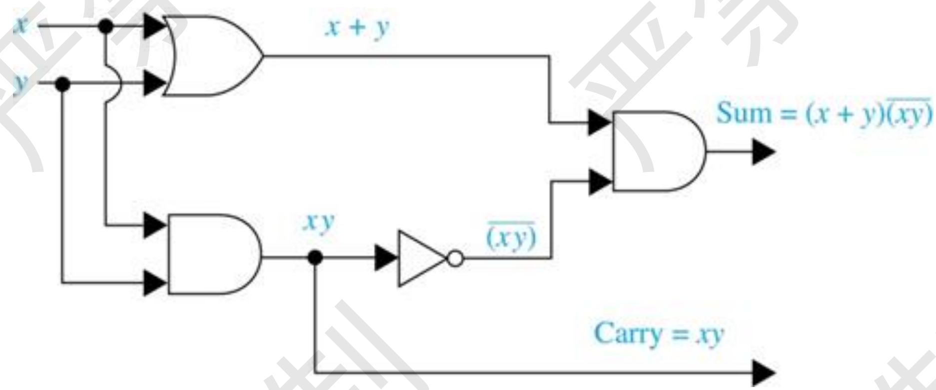


ADDERS

- Logic circuits can be used to add two positive integers from their binary expansions.
- The first step is to build a *half adder* that adds two bits, but which does not accept a carry from a previous addition.
- Since the circuit has more than one output, it is a *multiple output circuit*.

TABLE 3 Input and Output for the Half Adder.

Input		*Output*	
x	y	s	c
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0



$s = x\bar{y} + \bar{x}y = (x + y)(\overline{xy})$
$c = xy$

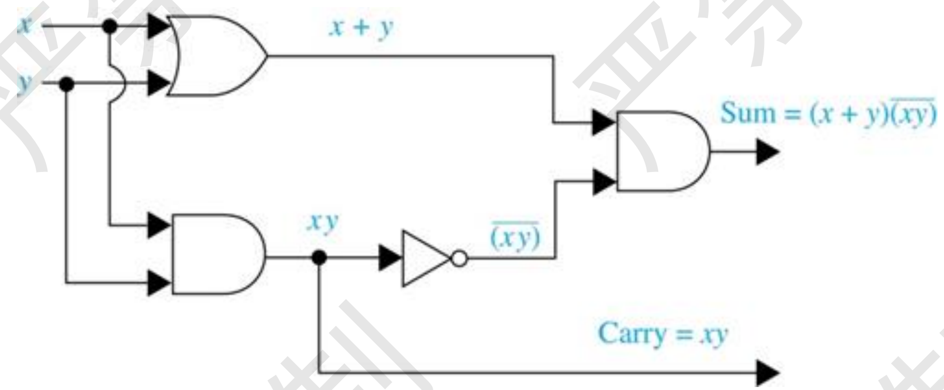
DNF

ADDERS

- Logic circuits can be used to add two positive integers from their binary expansions.
- The first step is to build a *half adder* that adds two bits, but which does not accept a carry from a previous addition.
- Since the circuit has more than one output, it is a *multiple output circuit*.

TABLE 3
Input and Output for the Half Adder.

Input		*Output*	
x	y	s	c
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0



$$s = x\overline{y} + \overline{x}y = (x + y)\overline{(xy)}$$
$$c = xy$$

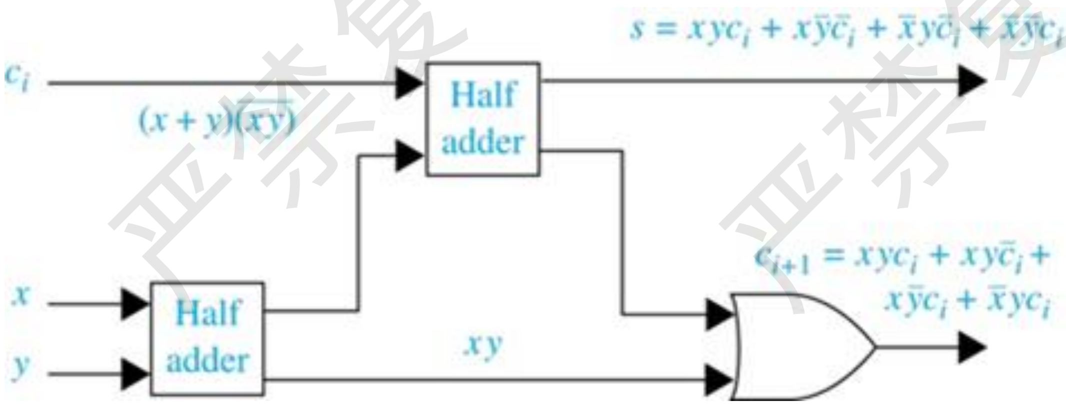
DNF

ADDERS (CONT)

- A *full adder* is used to compute the sum bit and the carry bit when two bits and a carry are added.

TABLE 4 Input and Output for the Full Adder.

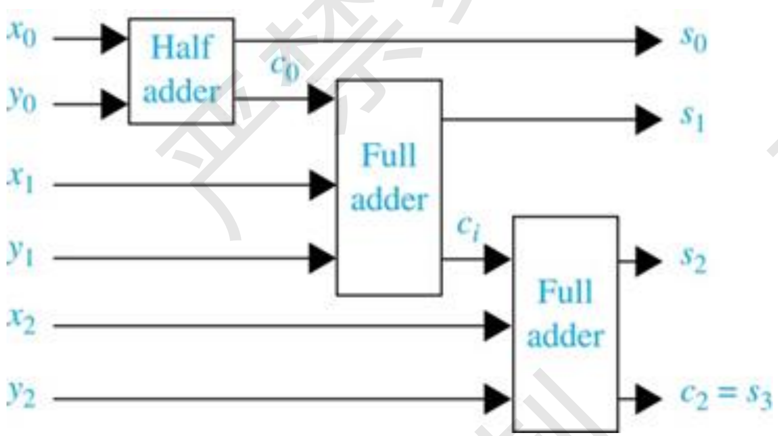
Input			Output	
x	y	c_i	s	c_{i+1}
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0



ADDERS (CONT)

- A half adder and multiple full adders can be used to produce the sum of *n* bit integers.

  Example: Here is a circuit to compute the sum of two three-bit integers.



THANK YOU !