Sec 1.1-1.2-1.3 Miscellaneous Applications

Instructor: Robert Mearns

1. Logic puzzle. We will do this problem two ways

Ex 1: Knights always tell the truth and Knaves always tell lies. There are two people A, B.
A says "Both of us are Knaves". B says nothing. Determine the type of A, B if possible.

Method 1: Reason without logic equivalences:

- Assume A is Knight. We have a contraction. Why ?
- Since A is a Knave what is B and Why ?

Method 2: It is not always easy to reason as in Method 1 hence: (i) translate into logic variables and operations then simplify using logic equivalences if possible (ii) use truth values:

Step 1 Let p represent "A is a Knight", q represent "B is a Knight":

Because of what A said we have the logic statement: $\mathbf{[p \to (\neg p \wedge \neg q)] \wedge [\neg p \to \neg(\neg p \wedge \neg q)]}$

$\mathbf{p \to (\neg p \wedge \neg q) \quad \wedge \quad \neg p \to \neg(\neg p \wedge \neg q)}$

Given

Conditional in terms of Or

$x \vee (x \wedge y) \cong x$, De Morgan

Associative , $x \vee x \cong x$

Distributive

Negation $(\neg x \wedge x) \cong F$

$x \vee F \cong x$

Step 2 Using truth values $\neg p \wedge q$ is Satisfied (is T) when $\neg p$ is T and B is T hence:

Note: If we start with: let p represent "A is Knave", q represent "B is Knight" the logic statement is different but will be its equivalent will give the same conclusion.

Sec 1.1-1.2-1.3 Miscellaneous Applications

Instructor: Robert Mearns

1. Logic puzzle. We will do this problem two ways

Ex 1: Knights always tell the truth and Knaves always tell lies. There are two people A, B. A says "Both of us are Knaves". B says nothing. Determine the type of A, B if possible.

<u>Method</u> 1: Reason without logic equivalences:

- Assume A is Knight. We have a contraction. Why ? Hence A is a Knave.
- Since A is a Knave what is B and Why ? Hence B is a Knight.

<u>Method 2</u>: It is not always easy to reason as in Method 1 hence: (i) translate into logic variables and operations then simplify using logic equivalences if possible (ii) use truth values:

Step 1 Let p represent "A is a Knight", q represent "B is a Knight":

Because of what A said we have the logic statement: $\mathbf{[p \to (\neg p \wedge \neg q)] \wedge [\neg p \to \neg(\neg p \wedge \neg q)]}$

$\mathbf{p \to (\neg p \wedge \neg q) \quad \wedge \quad \neg p \to \neg(\neg p \wedge \neg q)}$	Given
$\cong [\neg p \vee (\neg p \wedge \neg q)] \quad \wedge \quad [p \vee \neg(\neg p \wedge \neg q)]$	Conditional in terms of Or
$\cong \neg p \quad \wedge \quad [p \vee (p \vee q)]$	$x \vee (x \wedge y) \cong x$, De Morgan
$\cong \neg p \quad \wedge \quad (p \vee q)$	Associative , $x \vee x \cong x$
$\cong (\neg p \wedge p) \vee (\neg p \wedge q)$	Distributive
$\cong F \vee (\neg p \wedge q)$	Negation $(\neg x \wedge x) \cong F$
$\cong \neg p \wedge q$	$x \vee F \cong x$

Step 2 Using truth values $\neg p \wedge q$ is Satisfied (is T) when $\neg p$ is T and B is T hence: A is Knave, B is Knight

Note: If we start with: let p represent "A is Knave", q represent "B is Knight" the logic statement is different but will be its equivalent will give the same conclusion.

Ex 2: Using the Knights and Knaves definitions of Ex 1:

A says "At least one of us is a Knight". B says nothing. Determine the type of A, B if possible.

Step 1 Let p represent "A is a Knight", q represent "B is a Knight":

Because of what A said we have the logic statement: $\mathbf{[p \to (p \lor q)] \land [\neg p \to \neg (p \lor q)]}$

$\mathbf{[p \to (p \lor q)] \quad \land \quad [\neg p \to \neg (p \lor q)]}$

Given

Conditional in terms of Or

Assoc. , De Morgan

$\neg x \lor x \equiv T$, Distributive

$T \lor x \equiv T$, $\neg x \lor x \equiv T$

$T \land x \equiv x$

$T \land x \equiv x$

Step 2

When is $p \lor \neg q$ satisfied (is True) ?

p	**q**	**¬ q**	**p ∨ ¬ q**
T	T	F	
T	F	T	
F	T	F	
F	F	T	

Ex 2: Using the Knights and Knaves definitions of Ex 1:

A says "At least one of us is a Knight". B says nothing. Determine the type of A, B if possible.

Step 1 Let p represent "A is a Knight", q represent "B is a Knight":

Because of what A said we have the logic statement: $\mathbf{[p \rightarrow (p \vee q)] \wedge [\neg p \rightarrow \neg (p \vee q)]}$

$\mathbf{[p \rightarrow (p \vee q)]}$	$\wedge$	$\mathbf{[\neg p \rightarrow \neg (p \vee q)]}$	Given
$\neg p \vee (p \vee q)$	$\wedge$	$[p \vee \neg (p \vee q)]$	Conditional in terms of Or
$(\neg p \vee p) \vee q)$	$\wedge$	$[p \vee (\neg p \wedge \neg q)]$	Assoc. , De Morgan
$T \quad \vee q$	$\wedge$	$(p \vee \neg p) \wedge (p \vee \neg q)$	$\neg x \vee x \equiv T$, Distributive
T	$\wedge$	$T \quad \wedge (p \vee \neg q)]$	$T \vee x \equiv T$, $\neg x \vee x \equiv T$
T	$\wedge$	$p \vee \neg q$	$T \wedge x \equiv x$
	$p \vee \neg q$		$T \wedge x \equiv x$

Step 2

When is $p \vee \neg q$ satisfied (is True) ?

p	**q**	$\mathbf{\neg q}$	$\mathbf{p \vee \neg q}$
T	T	F	T
T	F	T	T
~~F~~	~~T~~	~~F~~	~~F~~
F	F	T	T

Three possibilities:

A Knight, B Knight
or A Knight, B Knave
or A Knave, B Knave

2. System specifications can be converted into precise specifications using logic variables to see if the specifications are Consistent: (Is there a set of truth values that make all specifications true).

Ex 1: Consider the specifications: - The error message is stored in the buffer or it is transmitted
- The error message is not stored in the buffer

Determine if the specifications are Consistent using logic equivalences

Step 1 Symbolize: Let p represent: The error message is stored in the buffer
Let q represent: The error message is transmitted:
We need to determine if there is a set of Truth values that makes:
(i) [$\mathbf{p \vee q}$ True and (ii) $\mathbf{\neg p}$ True] $\equiv$ $\mathbf{(p \vee q) \wedge \neg p}$ True

$\mathbf{(p \vee q) \wedge \neg p}$	Given
	Distributive
	$x \wedge \neg x \equiv F$
	$F \vee x \equiv x$

Step 2 Using truth values $\neg p \wedge q$ is Satisfied (is T) when $\neg p$ is T and B is T hence Consistent:

Ex 2: Add the specification "If the error message is not stored in the buffer then it is not transmitted" to the 2 given specifications in Ex 1. Symbolize and simplify the three specifications taken together with And. What is conclusion about the 3 specifications together ?

$\mathbf{(p \vee q) \wedge \neg p \wedge (\neg p \rightarrow \neg q)}$

	Ans Ex 1, conditional with or
	Distributive
	$\neg x \wedge x \equiv F$
	$F \wedge x \equiv F$
	$F \wedge F \equiv F$

2. System specifications can be converted into precise specifications using logic variables to see if the specifications are Consistent: (Is there a set of truth values that make all specifications true).

Ex 1: Consider the specifications:
- The error message is stored in the buffer or it is transmitted
- The error message is not stored in the buffer

Determine if the specifications are Consistent using logic equivalences

Step 1 Symbolize: Let p represent: The error message is stored in the buffer
Let q represent: The error message is transmitted:
We need to determine if there is a set of Truth values that makes:
(i) [$\mathbf{p \vee q}$ True and (ii) $\mathbf{\neg p}$ True] $\equiv$ $\mathbf{(p \vee q) \wedge \neg p}$ True

$\mathbf{(p \vee q) \wedge \neg p}$	Given
$\equiv (p \wedge \neg p) \vee (q \wedge \neg p)$	Distributive
$\equiv F \vee (q \wedge \neg p)$	$x \wedge \neg x \equiv F$
$\equiv q \wedge \neg p$	$F \vee x \equiv x$

Step 2 Using truth values $\neg p \wedge q$ is Satisfied (is T) when $\neg p$ is T and B is T hence Consistent:
Given specifications cause: Error message not stored in buffer and error message transmitted

Ex 2: Add the specification "If the error message is not stored in the buffer then it is not transmitted" to the 2 given specifications in Ex 1. Symbolize and simplify the three specifications taken together with And. What is conclusion about the 3 specifications together ?

$\mathbf{(p \vee q) \wedge \neg p \wedge (\neg p \rightarrow \neg q)}$

$\equiv (\neg p \wedge q) \wedge (p \vee \neg q)$	Ans Ex 1, conditional with or
$\equiv (\neg p \wedge q \wedge p) \vee (\neg p \wedge q \wedge \neg q)$	Distributive
$\equiv (F \wedge q) \vee (\neg p \wedge F)$	$\neg x \wedge x \equiv F$
$\equiv F \vee F$	$F \wedge x \equiv F$
$\equiv F$	$F \vee F \equiv F$

Conclusion: No truth values make all specifications True
Hence: Inconsistent

Ex 3: - Whenever system software is being upgraded users cannot access the file system

- If users can access the file system then they can save new files
- If users cannot save new files then the system software is not being upgraded

Determine if the specifications are Consistent using symbols and truth values

Symbolize: Let p represent: System software is being upgraded
Let q represent: Users can access the file system
Let s represent: Users can save new files

We need to determine if there is a set of Truth values for p, q, s that make the following true:

Consider:

(i)	$q = F \rightarrow$	Negation
	$(p = T) \wedge (\neg q = T) \rightarrow$	Truth table for Conditional
(ii)	$s = T \rightarrow$	Negation
	$p = T \rightarrow$	Negation
	$(\neg s = F) \wedge (\neg p = F) \rightarrow$	Truth table for Conditional
(iii)	$(q = F) \wedge (s = T) \rightarrow$	Truth table for Conditional
	$\rightarrow$	Using results of (i), (ii), (iii)

Ex 3: - Whenever system software is being upgraded users cannot access the file system

- If users can access the file system then they can save new files
- If users cannot save new files then the system software is not being upgraded

Determine if the specifications are Consistent using symbols and truth values

Symbolize: Let p represent: System software is being upgraded

Let q represent: Users can access the file system

Let s represent: Users can save new files

We need to determine if there is a set of Truth values for p, q, s that make the following true:

(i) $\mathbf{p \rightarrow \neg q}$ (ii) $\mathbf{q \rightarrow s}$ (iii) $\mathbf{\neg s \rightarrow \neg p}$

Consider: p = T , q = F , s = T

(i)	$q = F \rightarrow \neg q = T$	Negation
	$(p = T) \wedge (\neg q = T) \rightarrow$ $\mathbf{[p \rightarrow \neg q] = T}$	Truth table for Conditional
(ii)	$s = T \rightarrow \neg s = F$	Negation
	$p = T \rightarrow \neg p = F$	Negation
	$(\neg s = F) \wedge (\neg p = F) \rightarrow$ $\mathbf{[\neg s \rightarrow \neg p] = T}$	Truth table for Conditional
(iii)	$(q = F) \wedge (s = T) \rightarrow$ $\mathbf{[q \rightarrow s] = T}$	Truth table for Conditional
	$\rightarrow$ Specifications are Consistent	Using results of (i), (ii), (iii)

Note: There are other truth values that will make the three specifications true in this ex.

Check: p = F , q = F , s = F

p = F , q = T , s = T

p = F , q = F , s = T

2. Boolean Algebra (Invented by George Boole - 1854)

a) The fundamental memory unit in a digital computer circuit is called a Bit. It can have only two states:

b) These bits can be configured to store various data types. You declare which type of storage is to be used in the program code:

c) We look at Boolean types and the Boolean Algebra used to represent them:
$1 \equiv$ True $\quad\quad 0 \equiv$ False

The truth values are now easier to calculate. The only one that deviates from regular integer arithmetic is 1+1.

Ex 1: Write the Boolean table for p + q. (equivalent to truth table for $p \vee q$)

p	q	p + q
1	1	
1	0	
0	1	
0	0	

In a similar way tables can be constructed for: Not, $*$, $\oplus$, $\rightarrow$, $\leftrightarrow$

2. Boolean Algebra (Invented by George Boole - 1854)

a) The fundamental memory unit in a digital computer circuit is called a <u>Bit</u>. It can have only two states: power on, represented by 1 and power off, represented by 0.

b) These bits can be configured to store various data types. You declare which type of storage is to be used in the program code:

Logic (Boolean) variables	Integers	Real numbers	Character codes
0 ≡ False, 1 ≡ True	….-2,-1,0,1,2…..	0.357 x 10 E 5	A = 01000001

c) We look at Boolean types and the Boolean Algebra used to represent them:

1 ≡ True 0 ≡ False

Not p $\equiv \overline{p} \equiv \neg p$

$p * q \equiv p\,q \equiv p \wedge q$

$p + q \equiv p \vee q$

The truth values are now easier to calculate. The only one that deviates from regular integer arithmetic is 1+1.

This means $T \vee T \equiv T$. Hence 1+1 = 1.

Ex 1: Write the Boolean table for p + q. (equivalent to truth table for $p \vee q$)

p	q	p + q
1	1	1 +1 = 1
1	0	1 + 0 = 1
0	1	0 + 1 = 1
0	0	0 + 0 = 0

In a similar way tables can be constructed for: Not, $*$, $\oplus$, $\rightarrow$, $\leftrightarrow$

Ex 2: Consider the bit strings 1101 0001, 1000 1010, 1101 1110

Combine the corresponding bit positions with:

(i) +

(ii) $*$

(iii) $\oplus$

1101 0001

1000 1010

1101 1110

(i) Bitwise Or

(ii) Bitwise And

(iii) Bitwise Exclusive Or

Ex 2: Consider the bit strings 1101 0001, 1000 1010, 1101 1110

Combine the corresponding bit positions with:

(i) +

(ii) $*$

(iii) $\oplus$

		1101 0001
		1000 1010
		1101 1110
(i)	Bitwise Or	1101 1111
(ii)	Bitwise And	1000 0000
(iii)	Bitwise Exclusive Or	1000 0101

Ex 3: Consider the following Decision table. A blank means the variable can be either true or false and hence it is not included in the original equations.

(i) Write the Boolean equation at the bottom of each column
(each column equation requires all Boolean conditions in its column → And)

(ii) Write the final Boolean equation for the complete table
(final decision requires the Boolean conditions for Rule 1 or Rule 2 or Rule 3 or Rule 4 → Or)

(iii) Simplify the final Boolean equation.

	Rule 1	Rule 2	Rule 3	Rule 4
p	1	1		
q			1	0
r	0	1	1	1
s	1	1		

(i)

(ii)

(iii)

Ex 3: Consider the following Decision table. A blank means the variable can be either true or false and hence it is not included in the original equations.

(i) Write the Boolean equation at the bottom of each column
(each column equation requires all Boolean conditions in its column → And)
(ii) Write the final Boolean equation for the complete table
(final decision requires the Boolean conditions for Rule 1 or Rule 2 or Rule 3 or Rule 4 → Or)
(iii) Simplify the final Boolean equation.

	Rule 1	Rule 2	Rule 3	Rule 4
p	1	1		
q			1	0
r	0	1	1	1
s	1	1		

(i) $p\ \bar{r}\ s$ $\quad$ $p\ r\ s$ $\quad$ $q\ r$ $\quad$ $\bar{q}\ r$

(ii) $p\,\bar{r}\,s + p\,r\,s + q\,r + \bar{q}\,r$

(iii) $p\,s\,(r + \bar{r}) + r\,(q + \bar{q})$

$p\,s\,(1) + r\,(1)$

$p\,s + r$

Hence the 4 rules are equivalent to:
(p true and s true) or r true

Note: q need not be tested when evaluating the truth value of this Decision table