CompSci 102
Discrete Math for Computer Science

January 19, 2012

Prof. Rodger

Most Slides are modified from Rosen
Announcements

- Read for next time Chap. 1.4-1.6
- Recitation 1 is tomorrow
- Homework will be posted by Friday

- Today more logic
Classwork problem from last time

Each inhabitant of a remote village always tells the truth or always lies. A villager will only give "yes" or "no" response to a question a tourist asks.

Suppose you are a tourist visiting this area and come to a fork in the road. One branch leads to the ruins you want to visit; the other leads deep into the jungle.

A villager is standing at the fork in the road. What one question can you ask the villager to determine which branch to take?
Precedence of Logical operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>¬</td>
<td>1</td>
</tr>
<tr>
<td>∧</td>
<td>2</td>
</tr>
<tr>
<td>∨</td>
<td>3</td>
</tr>
<tr>
<td>→</td>
<td>4</td>
</tr>
<tr>
<td>↔</td>
<td>5</td>
</tr>
</tbody>
</table>

Example: $p \lor \neg q \land r \rightarrow s \lor q$
Precedence of Logical operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \neg )</td>
<td>1</td>
</tr>
<tr>
<td>( \land )</td>
<td>2</td>
</tr>
<tr>
<td>( \lor )</td>
<td>3</td>
</tr>
<tr>
<td>( \rightarrow )</td>
<td>4</td>
</tr>
<tr>
<td>( \leftrightarrow )</td>
<td>5</td>
</tr>
</tbody>
</table>

Example: \( p \lor \neg q \land r \rightarrow s \lor q \)

\[ (p \lor (\neg q \land r)) \rightarrow (s \lor q) \]
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.”
  – \( p \): I go to Harry’s
  – \( q \): I go to the country.
  – \( r \): I will go shopping.
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.”
  – $p$: I go to Harry’s
  – $q$: I go to the country.
  – $r$: I will go shopping.
  
  \[(p \lor q) \rightarrow \neg r\]
Example

Problem: Translate the following sentence into propositional logic:

“You can access the Internet from campus only if you are a computer science major or you are not a freshman.”
Example

Problem: Translate the following sentence into propositional logic:

“You can access the Internet from campus only if you are a computer science major or you are not a freshman.”

One Solution: Let $a$, $c$, and $f$ represent respectively “You can access the internet from campus,” “You are a computer science major,” and “You are a freshman.”

$$a \rightarrow (c \lor \neg f)$$
System Specifications

• System and Software engineers take requirements in English and express them in a precise specification language based on logic. **Example**: Express in propositional logic:

  “The automated reply cannot be sent when the file system is full”

  **Solution**: One possible solution: Let \( p \) denote “The automated reply can be sent” and \( q \) denote “The file system is full.”
System Specifications

- System and Software engineers take requirements in English and express them in a precise specification language based on logic.

Example: Express in propositional logic:
“The automated reply cannot be sent when the file system is full”

Solution: One possible solution: Let $p$ denote “The automated reply can be sent” and $q$ denote “The file system is full.”

$q \rightarrow \neg p$
Consistent System Specifications

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.

Exercise: Are these specifications consistent?

- “The diagnostic message is stored in the buffer or it is retransmitted.”
- “The diagnostic message is not stored in the buffer.”
- “If the diagnostic message is stored in the buffer, then it is retransmitted.”
- What if “The diagnostic message is not retransmitted is added.”
Consistent System Specifications

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.

Exercise: Are these specifications consistent?

- “The diagnostic message is stored in the buffer or it is retransmitted.”
- “The diagnostic message is not stored in the buffer.”
- “If the diagnostic message is stored in the buffer, then it is retransmitted.”

Solution: Let $p$ denote “The diagnostic message is not stored in the buffer.” Let $q$ denote “The diagnostic message is retransmitted” The specification can be written as: $p \lor q$, $p \rightarrow q$, $\neg p$. When $p$ is false and $q$ is true all three statements are true. So the specification is consistent.

- What if “The diagnostic message is not retransmitted is added.”

Solution: Now we are adding $\neg q$ and there is no satisfying assignment. So the specification is not consistent.
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 “The two of us are both knights”
  – B says “A is a Knave.”

Example: What are the types of A and B?
Tautologies, Contradictions, and Contingencies

• A tautology is a proposition which is always true.
  – Example: $p \lor \neg p$

• A contradiction is a proposition which is always false.
  – Example: $p \land \neg p$

• A contingency is a proposition which is neither a tautology nor a contradiction, such as $p$

<table>
<thead>
<tr>
<th>$P$</th>
<th>$\neg P$</th>
<th>$P \lor \neg P$</th>
<th>$P \land \neg P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>
Logically Equivalent

- Two compound propositions $p$ and $q$ are logically equivalent if $p \leftrightarrow q$ is a tautology.
- We write this as $p \equiv q$ or as $p \equiv q$ where $p$ and $q$ are compound propositions.
- Two compound propositions $p$ and $q$ are equivalent if and only if the columns in a truth table giving their truth values agree.
- This truth table show $\neg p \lor q$ is equivalent to $p \rightarrow q$.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$\neg p$</th>
<th>$\neg p \lor q$</th>
<th>$p \rightarrow q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
De Morgan’s Laws

\[ \neg(p \land q) \equiv \neg p \lor \neg q \]

\[ \neg(p \lor q) \equiv \neg p \land \neg q \]

This truth table shows that De Morgan’s Second Law holds.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Key Logical Equivalences

• **Identity Laws:**
  \[ p \land T \equiv p \quad p \lor F \equiv p \]

• **Domination Laws:**
  \[ p \lor T \equiv T \quad p \land F \equiv F \]

• **Idempotent laws:**
  \[ p \lor p \equiv p \quad p \land p \equiv p \]

• **Double Negation Law:**
  \[ \neg(\neg p) \equiv p \]

• **Negation Laws:**
  \[ p \lor \neg p \equiv T \quad p \land \neg p \equiv F \]
Key Logical Equivalences (cont)

• Commutative Laws: \( p \lor q \equiv q \lor p \), \( p \land q \equiv q \land p \)

• Associative Laws: 
  \((p \land q) \land r \equiv p \land (q \land r)\)
  \((p \lor q) \lor r \equiv p \lor (q \lor r)\)

• Distributive Laws: 
  \((p \lor (q \land r)) \equiv (p \lor q) \land (p \lor r)\)
  \((p \land (q \lor r)) \equiv (p \land q) \lor (p \land r)\)

• Absorption Laws: \( p \lor (p \land q) \equiv p \) \( p \land (p \lor q) \equiv p \)
More Logical Equivalences

**TABLE 7** Logical Equivalences Involving Conditional Statements.

<table>
<thead>
<tr>
<th>Logical Equivalence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p \rightarrow q \equiv \neg p \vee q )</td>
</tr>
<tr>
<td>( p \rightarrow q \equiv \neg q \rightarrow \neg p )</td>
</tr>
<tr>
<td>( p \lor q \equiv \neg p \rightarrow q )</td>
</tr>
<tr>
<td>( p \land q \equiv \neg(p \rightarrow \neg q) )</td>
</tr>
<tr>
<td>( \neg(p \rightarrow q) \equiv p \land \neg q )</td>
</tr>
<tr>
<td>( (p \rightarrow q) \land (p \rightarrow r) \equiv p \rightarrow (q \land r) )</td>
</tr>
<tr>
<td>( (p \rightarrow r) \land (q \rightarrow r) \equiv (p \lor q) \rightarrow r )</td>
</tr>
<tr>
<td>( (p \rightarrow q) \lor (p \rightarrow r) \equiv p \rightarrow (q \lor r) )</td>
</tr>
<tr>
<td>( (p \rightarrow r) \lor (q \rightarrow r) \equiv (p \land q) \rightarrow r )</td>
</tr>
</tbody>
</table>

**TABLE 8** Logical Equivalences Involving Biconditional Statements.

<table>
<thead>
<tr>
<th>Logical Equivalence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p \leftrightarrow q \equiv (p \rightarrow q) \land (q \rightarrow p) )</td>
</tr>
<tr>
<td>( p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q )</td>
</tr>
<tr>
<td>( p \leftrightarrow q \equiv (p \land q) \lor (\neg p \land \neg q) )</td>
</tr>
<tr>
<td>( \neg(p \leftrightarrow q) \equiv p \leftrightarrow \neg q )</td>
</tr>
</tbody>
</table>
Constructing New Logical Equivalences

• We can show that two expressions are logically equivalent by developing a series of logically equivalent statements.

• To prove that $A \equiv B$ we produce a series of equivalences beginning with $A$ and ending with $B$.

$$A \equiv A_1 \\
\vdots \\
A_n \equiv B$$

• Keep in mind that whenever a proposition (represented by a propositional variable) occurs in the equivalences listed earlier, it may be replaced by an arbitrarily complex compound proposition.
Equivalence Proofs

Example: Show that \( \neg(p \lor (\neg p \land q)) \)

is logically equivalent to \( \neg p \land \neg q \)
Equivalence Proofs

Example: Show that $\neg(p \lor (\neg p \land q))$ is logically equivalent to $\neg p \land \neg q$

Solution:

\[
\begin{align*}
\neg(p \lor (\neg p \land q)) & \equiv \neg p \land \neg (\neg p \land q) & \text{by the second De Morgan law} \\
& \equiv \neg p \land [\neg (\neg p) \lor \neg q] & \text{by the first De Morgan law} \\
& \equiv \neg p \land (p \lor \neg q) & \text{by the double negation law} \\
& \equiv (\neg p \land p) \lor (\neg p \land \neg q) & \text{by the second distributive law} \\
& \equiv F \lor (\neg p \land \neg q) & \text{because } \neg p \land p \equiv F \\
& \equiv (\neg p \land \neg q) \lor F & \text{by the commutative law for disjunction} \\
& \equiv (\neg p \land \neg q) & \text{by the identity law for } F
\end{align*}
\]
Equivalence Proofs

Example: Show that \((p \land q) \rightarrow (p \lor q)\) is a tautology.

Solution:
Equivalence Proofs

Example: Show that \((p \land q) \rightarrow (p \lor q)\)

is a tautology.

Solution:

\[
(p \land q) \rightarrow (p \lor q) \\
\equiv \neg(p \land q) \lor (p \lor q) \\
\equiv (\neg p \lor \neg q) \lor (p \lor q) \\
\equiv (\neg p \lor p) \lor (\neg p \lor \neg q) \\
\equiv T \lor T \\
\equiv T
\]

by truth table for \(\rightarrow\)

by the first De Morgan law

by associative and commutative laws

laws for disjunction

by truth tables

by the domination law
Propositional Satisfiability

- A compound proposition is *satisfiable* if there is an assignment of truth values to its variables that make it true. When no such assignments exist, the compound proposition is *unsatisfiable*.
- A compound proposition is unsatisfiable if and only if its negation is a tautology.
Questions on Propositional Satisfiability

**Example:** Determine the satisfiability of the following compound propositions:

\[(p \lor \neg q) \land (q \lor \neg r) \land (r \lor \neg p)\]

\[(p \lor q \lor r) \land (\neg p \lor \neg q \lor \neg r)\]

\[(p \lor \neg q) \land (q \lor \neg r) \land (r \lor \neg p) \land (p \lor q \lor r) \land (\neg p \lor \neg q \lor \neg r)\]
Questions on Propositional Satisfiability

Example: Determine the satisfiability of the following compound propositions:

\[(p \lor \neg q) \land (q \lor \neg r) \land (r \lor \neg p)\]

Solution: Satisfiable. Assign \(T\) to \(p, q,\) and \(r\).

\[(p \lor q \lor r) \land (\neg p \lor \neg q \lor \neg r)\]

Solution: Satisfiable. Assign \(T\) to \(p\) and \(F\) to \(q\).

\[(p \lor \neg q) \land (q \lor \neg r) \land (r \lor \neg p) \land (p \lor q \lor r) \land (\neg p \lor \neg q \lor \neg r)\]

Solution: Not satisfiable. Check each possible assignment of truth values to the propositional variables and none will make the proposition true.
Satisfiability problem

• First CS problem to be shown NP-Complete
  – Problems that take too much time to solve….
  – Cook 1971
  – Math professor at UC Berkeley – now U Toronto
• Start of the area: Complexity theory
• Many problems now shown NP-Complete
Notation

\[ \bigvee_{j=1}^{n} p_j \] is used for \( p_1 \lor p_2 \lor \ldots \lor p_n \)

\[ \bigwedge_{j=1}^{n} p_j \] is used for \( p_1 \land p_2 \land \ldots \land p_n \)

Needed for the next example.
Sudoku

• A **Sudoku puzzle** is represented by a $9 \times 9$ grid made up of nine $3 \times 3$ subgrids, known as **blocks**. Some of the 81 cells of the puzzle are assigned one of the numbers 1, 2, ..., 9.

• The puzzle is solved by assigning numbers to each blank cell so that every row, column and block contains each of the nine possible numbers.

• Example
Encoding as a Satisfiability Problem

- Let $p(i,j,n)$ denote the proposition that is true when the number $n$ is in the cell in the $i$th row and the $j$th column.
- There are $9 \times 9 \times 9 = 729$ such propositions.
- In the sample puzzle $p(5,1,6)$ is true, but $p(5,j,6)$ is false for $j = 2,3,\ldots,9$
Encoding (cont)

• For each cell with a given value, assert $p(d,j,n)$, when the cell in row $i$ and column $j$ has the given value.

• Assert that every row contains every number.
$$
\bigwedge_{i=1}^{9} \bigwedge_{n=1}^{9} \bigvee_{j=1}^{9} p(i, j, n)
$$

• Assert that every column contains every number.
$$
\bigwedge_{j=1}^{9} \bigwedge_{i=1}^{9} \bigvee_{n=1}^{9} p(i, j, n)
$$
Encoding (cont)

• Assert that each of the 3 x 3 blocks contain every number. $\bigwedge_{r=0}^{2} \bigwedge_{s=0}^{2} \bigwedge_{n=1}^{9} \bigwedge_{i=1}^{3} \bigwedge_{j=1}^{3} p(3r + i, 3s + j, n)$

• Assert that no cell contains more than one number. Take the conjunction over all values of $n, n', i$, and $j$, where each variable ranges from 1 to 9 and $n \neq n'$, of $p(i, j, n) \rightarrow \neg p(i, j, n')$
Solving Satisifiability Problems

• To solve a Sudoku puzzle, we need to find an assignment of truth values to the 729 variables of the form $p(i,j,n)$ that makes the conjunction of the assertions true. Those variables that are assigned T yield a solution to the puzzle.

• A truth table can always be used to determine the satisfiability of a compound proposition. But this is too complex even for modern computers for large problems.

• There has been much work on developing efficient methods for solving satisfiability problems as many practical problems can be translated into satisfiability problems.
Propositional Logic Not Enough

• If we have:
  “All men are mortal.”
  “Socrates is a man.”
• Does it follow that “Socrates is mortal?”
• Can’t be represented in propositional logic. Need a language that talks about objects, their properties, and their relations.
• Later we’ll see how to draw inferences.
Introducing Predicate Logic

• Predicate logic uses the following new features:
  – Variables: $x, y, z$
  – Predicates: $P(x), M(x)$
  – Quantifiers (*to be covered in a few slides*):

• *Propositional functions* are a generalization of propositions.
  – They contain variables and a predicate, e.g., $P(x)$
  – Variables can be replaced by elements from their *domain*. 
Propositional Functions

• Propositional functions become propositions (and have truth values) when their variables are each replaced by a value from the domain (or bound by a quantifier, as we will see later).

• The statement \( P(x) \) is said to be the value of the propositional function \( P \) at \( x \).

• For example, let \( P(x) \) denote “\( x > 0 \)” and the domain be the integers. Then:
  
  \[
  \begin{align*}
  P(-3) & \text{ is false.} \\
  P(0) & \text{ is false.} \\
  P(3) & \text{ is true.}
  \end{align*}
  \]

• Often the domain is denoted by \( U \). So in this example \( U \) is the integers.
Examples of Propositional Functions

• Let “$x + y = z$” be denoted by $R(x, y, z)$ and $U$ (for all three variables) be the integers. Find these truth values:
  $R(2,-1,5)$
  $R(3,4,7)$
  $R(x, 3, z)$

• Now let “$x - y = z$” be denoted by $Q(x, y, z)$, with $U$ as the integers. Find these truth values:
  $Q(2,-1,3)$
  $Q(3,4,7)$
  $Q(x, 3, z)$
Examples of Propositional Functions

• Let “\( x + y = z \)” be denoted by \( R(x, y, z) \) and \( U \) (for all three variables) be the integers. Find these truth values:
  \( R(2, -1, 5) \)
  Solution: F
  \( R(3, 4, 7) \)
  Solution: T
  \( R(x, 3, z) \)
  Solution: Not a Proposition

• Now let “\( x - y = z \)” be denoted by \( Q(x, y, z) \), with \( U \) as the integers. Find these truth values:
  \( Q(2, -1, 3) \)
  Solution: T
  \( Q(3, 4, 7) \)
  Solution: F
  \( Q(x, 3, z) \)
  Solution: Not a Proposition
Compound Expressions

• Connectives from propositional logic carry over to predicate logic.

• If $P(x)$ denotes “$x > 0$,” find these truth values:
  - $P(3) \lor P(-1)$
  - $P(3) \land P(-1)$
  - $P(3) \rightarrow P(-1)$
  - $P(3) \rightarrow P(-1)$

• Expressions with variables are not propositions and therefore do not have truth values. For example,
  - $P(3) \land P(y)$
  - $P(x) \rightarrow P(y)$

• When used with quantifiers (to be introduced next), these expressions (propositional functions) become propositions.
Compound Expressions

• Connectives from propositional logic carry over to predicate logic.

• If $P(x)$ denotes “$x > 0$,” find these truth values:
  
  \[ P(3) \lor P(-1) \quad \text{T} \]
  
  \[ P(3) \land P(-1) \quad \text{F} \]
  
  \[ P(3) \rightarrow P(-1) \quad \text{F} \]
  
  \[ P(3) \rightarrow P(-1) \quad \text{T} \]

• Expressions with variables are not propositions and therefore do not have truth values. For example,
  
  \[ P(3) \land P(y) \]
  
  \[ P(x) \rightarrow P(y) \]

• When used with quantifiers (to be introduced next), these expressions (propositional functions) become propositions.
Quantifiers

• We need quantifiers to express the meaning of English words including all and some:
  – “All men are Mortal.”
  – “Some cats do not have fur.”
• The two most important quantifiers are:
  – Universal Quantifier, “For all,” symbol: ∀
  – Existential Quantifier, “There exists,” symbol: ∃
• We write as in ∀x P(x) and ∃x P(x).
• ∀x P(x) asserts P(x) is true for every x in the domain.
• ∃x P(x) asserts P(x) is true for some x in the domain.
• The quantifiers are said to bind the variable x in these expressions.
Universal Quantifier

\( \forall x \, P(x) \) is read as "For all \( x \), \( P(x) \)" or "For every \( x \), \( P(x) \)"

Examples:

1) If \( P(x) \) denotes "\( x > 0 \)" and \( U \) is the integers, then \( \forall x \, P(x) \) is false.

2) If \( P(x) \) denotes "\( x > 0 \)" and \( U \) is the positive integers, then \( \forall x \, P(x) \) is true.

3) If \( P(x) \) denotes "\( x \) is even" and \( U \) is the integers, then \( \forall x \, P(x) \) is false.
Existential Quantifier

• $\exists x\ P(x)$ is read as “For some $x$, $P(x)$”, or as “There is an $x$ such that $P(x)$,” or “For at least one $x$, $P(x)$.”

Examples:

1. If $P(x)$ denotes “$x > 0$” and $U$ is the integers, then $\exists x\ P(x)$ is true. It is also true if $U$ is the positive integers.

2. If $P(x)$ denotes “$x < 0$” and $U$ is the positive integers, then $\exists x\ P(x)$ is false.

3. If $P(x)$ denotes “$x$ is even” and $U$ is the integers, then $\exists x\ P(x)$ is true.
Uniqueness Quantifier

• $\exists! x \; P(x)$ means that $P(x)$ is true for one and only one $x$ in the universe of discourse.

• This is commonly expressed in English in the following equivalent ways:
  – “There is a unique $x$ such that $P(x)$.”
  – “There is one and only one $x$ such that $P(x)$”

• Examples:
  1. If $P(x)$ denotes “$x + 1 = 0$” and $U$ is the integers, then $\exists! x \; P(x)$ is true.
  2. But if $P(x)$ denotes “$x > 0$,” then $\exists! x \; P(x)$ is false.

• The uniqueness quantifier is not really needed as the restriction that there is a unique $x$ such that $P(x)$ can be expressed as:

\[ \exists x \; (P(x) \land \forall y \; (P(y) \rightarrow y = x)) \]
Thinking about Quantifiers

• When the domain of discourse is finite, we can think of quantification as looping through the elements of the domain.

• To evaluate $\forall x \ P(x)$ loop through all $x$ in the domain.
  – If at every step $P(x)$ is true, then $\forall x \ P(x)$ is true.
  – If at a step $P(x)$ is false, then $\forall x \ P(x)$ is false and the loop terminates.

• To evaluate $\exists x \ P(x)$ loop through all $x$ in the domain.
  – If at some step, $P(x)$ is true, then $\exists x \ P(x)$ is true and the loop terminates.
  – If the loop ends without finding an $x$ for which $P(x)$ is true, then $\exists x \ P(x)$ is false.

• Even if the domains are infinite, we can still think of the quantifiers this fashion, but the loops will not terminate in some cases.
Properties of Quantifiers

• The truth value of $\exists x \ P(x)$ and $\forall x \ P(x)$ depend on both the propositional function $P(x)$ and on the domain $U$.

• Examples:
  1. If $U$ is the positive integers and $P(x)$ is the statement “$x < 2$”, then $\exists x \ P(x)$ is true, but $\forall x \ P(x)$ is false.
  2. If $U$ is the negative integers and $P(x)$ is the statement “$x < 2$”, then both $\exists x \ P(x)$ and $\forall x \ P(x)$ are true.
  3. If $U$ consists of 3, 4, and 5, and $P(x)$ is the statement “$x > 2$”, then both $\exists x \ P(x)$ and $\forall x \ P(x)$ are true. But if $P(x)$ is the statement “$x < 2$”, then both $\exists x \ P(x)$ and $\forall x \ P(x)$ are false.
Precedence of Quantifiers

• The quantifiers $\forall$ and $\exists$ have higher precedence than all the logical operators.

• For example, $\forall x P(x) \lor Q(x)$ means $(\forall x P(x)) \lor Q(x)$

• $\forall x (P(x) \lor Q(x))$ means something different.

• Unfortunately, often people write $\forall x P(x) \lor Q(x)$ when they mean $\forall x (P(x) \lor Q(x))$.  
Translating from English to Logic

Example 1: Translate the following sentence into predicate logic: “Every student in this class has taken a course in Java.”

Solution:

First decide on the domain $U$.

Solution 1: If $U$ is all students in this class, define a propositional function $J(x)$ denoting “$x$ has taken a course in Java” and translate as $\forall x J(x)$.

Solution 2: But if $U$ is all people, also define a propositional function $S(x)$ denoting “$x$ is a student in this class” and translate as $\forall x (S(x) \rightarrow J(x))$.

$\forall x (S(x) \land J(x))$ is not correct. What does it mean?
Translating from English to Logic

**Example 2:** Translate the following sentence into predicate logic: “Some student in this class has taken a course in Java.”

**Solution:**
First decide on the domain $U$.

**Solution 1:** If $U$ is all students in this class, translate as
$$\exists x \ J(x)$$

**Solution 1:** But if $U$ is all people, then translate as
$$\exists x \ (S(x) \land J(x))$$

$$\exists x \ (S(x) \rightarrow J(x))$$ is not correct. What does it mean?
Returning to the Socrates Example

• Introduce the propositional functions $\text{Man}(x)$ denoting “$x$ is a man” and $\text{Mortal}(x)$ denoting “$x$ is mortal.” Specify the domain as all people.

• The two premises are: $\forall x \text{Man}(x) \rightarrow \text{Mortal}(x)$
  \[
  \text{Man}(\text{Socrates})
  \]

• The conclusion is: $\text{Mortal}(\text{Socrates})$
Equivalences in Predicate Logic

• Statements involving predicates and quantifiers are *logically equivalent* if and only if they have the same truth value
  – for every predicate substituted into these statements and
  – for every domain of discourse used for the variables in the expressions.

• The notation $S \equiv T$ indicates that $S$ and $T$ are logically equivalent.

• **Example:** $\forall x \neg \neg S(x) \equiv \forall x S(x)$
Thinking about Quantifiers as Conjunctions and Disjunctions

• If the domain is finite, a universally quantified proposition is equivalent to a conjunction of propositions without quantifiers and an existentially quantified proposition is equivalent to a disjunction of propositions without quantifiers.
• If $U$ consists of the integers 1, 2, and 3:

$$\forall x P(x) \equiv P(1) \land P(2) \land P(3)$$

$$\exists x P(x) \equiv P(1) \lor P(2) \lor P(3)$$

• Even if the domains are infinite, you can still think of the quantifiers in this fashion, but the equivalent expressions without quantifiers will be infinitely long.
Negating Quantified Expressions

• Consider $\forall x J(x)$
  “Every student in your class has taken a course in Java.”
  Here $J(x)$ is “$x$ has taken a course in calculus” and the domain is students in your class.

• Negating the original statement gives “It is not the case that every student in your class has taken Java.”
  This implies that “There is a student in your class who has not taken calculus.”

  Symbolically $\neg \forall x J(x)$ and $\exists x \neg J(x)$ are equivalent
Negating Quantified Expressions (continued)

• Now Consider $\exists x J(x)$
  “There is a student in this class who has taken a course in Java.”
  Where $J(x)$ is “$x$ has taken a course in Java.”

• Negating the original statement gives “It is not the case that there is a student in this class who has taken Java.” This implies that “Every student in this class has not taken Java”
  Symbolically $\neg \exists x J(x)$ and $\forall x \neg J(x)$ are equivalent
De Morgan’s Laws for Quantifiers

- The rules for negating quantifiers are:

<table>
<thead>
<tr>
<th>Negation</th>
<th>Equivalent Statement</th>
<th>When Is Negation True?</th>
<th>When False?</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \neg \exists x P(x) )</td>
<td>( \forall x \neg P(x) )</td>
<td>For every ( x ), ( P(x) ) is false.</td>
<td>There is an ( x ) for which ( P(x) ) is true.</td>
</tr>
<tr>
<td>( \neg \forall x P(x) )</td>
<td>( \exists x \neg P(x) )</td>
<td>There is an ( x ) for which ( P(x) ) is false.</td>
<td>( P(x) ) is true for every ( x ).</td>
</tr>
</tbody>
</table>

- The reasoning in the table shows that:

  \[ \neg \forall x P(x) \equiv \exists x \neg P(x) \]

  \[ \neg \exists x P(x) \equiv \forall x \neg P(x) \]

- These are important. You will use these.
Some Fun with Translating from English into Logical Expressions

- \( U = \{ \text{fleegles, snurds, thingamabobs} \} \)
  - \( F(x) \): \( x \) is a fleegle
  - \( S(x) \): \( x \) is a snurd
  - \( T(x) \): \( x \) is a thingamabob

Translate “Everything is a fleegle”

**Solution:** \( \forall x \ F(x) \)
Translation (cont)

- $U = \{\text{fleegles, snurds, thingamabobs}\}$
  
  $F(x): x$ is a fleegle
  
  $S(x): x$ is a snurd
  
  $T(x): x$ is a thingamabob

  “Nothing is a snurd.”

  **Solution:** $\neg \exists x \ S(x)$  
  What is this equivalent to?

  **Solution:** $\forall x \neg S(x)$
Translation (cont)

• \( U = \{ \text{fleegles, snurds, thingamabobs} \} \)
  \( F(x) \): \( x \) is a fleegle
  \( S(x) \): \( x \) is a snurd
  \( T(x) \): \( x \) is a thingamabob

  “All fleegles are snurds.”

**Solution:** \( \forall x \ (F(x) \rightarrow S(x)) \)
Translation (cont)

- $U = \{ \text{fleegles, snurds, thingamabobs} \}$
  - $F(x)$: $x$ is a fleegle
  - $S(x)$: $x$ is a snurd
  - $T(x)$: $x$ is a thingamabob

“Some fleegles are thingamabobs.”

Solution: $\exists x \ (F(x) \land T(x))$
Translation (cont)

• \( U = \{ \text{fleegles, snurds, thingamabobs} \} \)
  
  \[ F(x): x \text{ is a fleegle} \]
  
  \[ S(x): x \text{ is a snurd} \]
  
  \[ T(x): x \text{ is a thingamabob} \]
  
  “No snurd is a thingamabob.”

Solution: \( \neg \exists x \left( S(x) \wedge T(x) \right) \) What is this equivalent to?

Solution: \( \forall x \left( \neg S(x) \lor \neg T(x) \right) \)
Translation (cont)

- $U = \{\text{fleegles, snurds, thingamabobs}\}$
  
  $F(x)$: $x$ is a fleegle
  
  $S(x)$: $x$ is a snurd
  
  $T(x)$: $x$ is a thingamabob

  “If any fleegle is a snurd then it is also a thingamabob.”

Solution: $\forall x ((F(x) \land S(x)) \rightarrow T(x))$