to be valid if its conclusion is true in cases where its premises are also true. Hence, a valid set of statements such as the ones above can give a false conclusion, provided one or more of the premises are also false.

We can say: a piece of reasoning is valid if it leads to a true conclusion in every situation where the premises are true.

Logic is concerned with truth values. The possible truth values are true and false. These can be considered to be the fundamental units of logic, and almost all logic is ultimately concerned with these truth values.

Logic is widely used in computer science, and particularly in Artificial Intelligence. Logic is widely used as a representational method for Artificial Intelligence. Unlike some other representations, logic allows us to easily reason about negatives (such as, “this book is not red”) and disjunctions (“or”—such as, “He’s either a soldier or a sailor”).

Logic is also often used as a representational method for communicating concepts and theories within the Artificial Intelligence community. In addition, logic is used to represent language in systems that are able to understand and analyze human language.

As we will see, one of the main weaknesses of traditional logic is its inability to deal with uncertainty. Logical statements must be expressed in terms of truth or falsehood—it is not possible to reason, in classical logic, about possibilities. We will see different versions of logic such as modal logics that provide some ability to reason about possibilities, and also probabilistic methods and fuzzy logic that provide much more rigorous ways to reason in uncertain situations.

Logical Operators

- In reasoning about truth values, we need to use a number of operators, which can be applied to truth values.
- We are familiar with several of these operators from everyday language:

  I like apples and oranges.

  You can have an ice cream or a cake.

If you come from France, then you speak French.

- Here we see the four most basic logical operators being used in everyday language. The operators are:
  - and
  - or
  - not
  - if . . . then . . . (usually called implies)
- One important point to note is that or is slightly different from the way we usually use it. In the sentence, “You can have an icecream or a cake,” the mother is usually suggesting to her child that he can only have one of the items, but not both. This is referred to as an exclusive-or in logic because the case where both are allowed is excluded.
- The version of or that is used in logic is called inclusive-or and allows the case with both options.
- The operators are usually written using the following symbols, although other symbols are sometimes used, according to the context:

and $\wedge$

or $\vee$

not $\neg$

implies $\rightarrow$

iff $\leftrightarrow$

- Iff is an abbreviation that is commonly used to mean “if and only if.”
- We see later that this is a stronger form of implies that holds true if one thing implies another, and also the second thing implies the first.
- For example, “you can have an ice-cream if and only if you eat your dinner.” It may not be immediately apparent why this is different from “you can have an icecream if you eat your dinner.” This is because most mothers really mean iff when they use if in this way.

Translating between English and Logic Notation

- To use logic, it is first necessary to convert facts and rules about the real world into logical expressions using the logical operators
- Without a reasonable amount of experience at this translation, it can seem quite a daunting task in some cases.
- Let us examine some examples. First, we will consider the simple operators, $\wedge$, $\vee$, and $\neg$.
- Sentences that use the word and in English to express more than one concept, all of which is true at once, can be easily translated into logic using the AND operator, $\wedge$.
- For example: "It is raining and it is Tuesday." might be expressed as: $R \wedge T$, Where R means "it is raining" and T means "it is Tuesday."
- For example, if it is not necessary to discuss where it is raining, R is probably enough.
- If we need to write expressions such as "it is raining in New York" or "it is raining heavily" or even "it rained for 30 minutes on Thursday," then R will probably not suffice. To express more complex concepts like these, we usually use predicates. Hence, for example, we might translate "it is raining in New York" as: N(R) We might equally well choose to write it as: R(N)
- This depends on whether we consider the rain to be a property of New York, or vice versa. In other words, when we write N(R), we are saying that a property of the rain is that it is in New York, whereas with R(N) we are saying that a property of New York is that it is raining. Which we use depends on the problem we are solving. It is likely that if we are solving a problem about New York, we would use R(N), whereas if we are solving a problem about the location of various types of weather, we might use N(R).
- Let us return nowto the logical operators. The expression "it is raining inNew York, and I'meither getting sick or just very tired"can be expressed as follows: $R(N) \wedge (S(I) \vee T(I))$
- Here we have used both the $\wedge$ operator, and the $\vee$ operator to express a collection of statements. The statement can be broken down into two sections, which is indicated by the use of parentheses.

- The section in the parentheses is $S(I) \vee T(I)$, which means “I’m either getting sick OR I’m very tired”. This expression is “AND’ed”with the part outside the parentheses, which is $R(N)$.
- Finally, the $\neg$ operator is applied exactly as you would expect—to express negation.
- For example, It is not raining in New York, might be expressed as $\neg R(N)$
- It is important to get the $\neg$ in the right place. For example: “I’m either not well or just very tired” would be translated as $\neg W(I) \vee T(I)$
- The position of the $\neg$ here indicates that it is bound to $W(I)$ and does not play any role in affecting $T(I)$.
- Now let us see how the →□operator is used. Often when dealing with logic we are discussing rules, which express concepts such as “if it is raining then I will get wet.”
- This sentence might be translated into logic as $R \rightarrow W(I)$
- This is read “R implies W(I)” or “IF R THEN W(I)”. By replacing the symbols R and W(I) with their respective English language equivalents, we can see that this sentence can be read as “raining implies I’ll get wet” or “IF it’s raining THEN I’ll get wet.”
- Implication can be used to express much more complex concepts than this.
- For example, “Whenever he eats sandwiches that have pickles in them, he ends up either asleep at his desk or singing loud songs” might be translated as

$$S(y) \wedge E(x, y) \wedge P(y) \rightarrow A(x) \vee (S(x, z) \wedge L(z))$$

- Here we have used the following symbol translations: $S(y)$ means that y is a sandwich. $E(x, y)$ means that x (the man) eats y (the sandwich).

  $P(y)$ means that y (the sandwich) has pickles in it.

  $A(x)$ means that x ends up asleep at his desk.

  $S(x, z)$ means that x (the man) sings z (songs).

  $L(z)$ means that z (the songs) are loud.

- The important thing to realize is that the choice of variables and predicates is important, but that you can choose any variables and predicates that map well to your problem and that help you to solve the problem.
- For example, in the example we have just looked at, we could perfectly well have used instead $S \rightarrow A \vee L$ where S means “he eats a sandwich which has pickles in it,” A means “he ends up asleep at his desk,” and L means “he sings loud songs.”
- The choice of granularity is important, but there is no right or wrong way to make this choice. In this simpler logical expression, we have chosen to express a simple relationship between three variables, which makes sense if those variables are all that we care about—in other words, we don’t need to know anything else about the sandwich, or the songs, or the man, and the facts we examine are simply whether or not he eats a sandwich with pickles, sleeps at his desk, and sings loud songs.
- The first translation we gave is more appropriate if we need to examine these concepts in more detail and reason more deeply about the entities involved.
- Note that we have thus far tended to use single letters to represent logical variables. It is also perfectly acceptable to use longer variable names, and thus to write expressions such as the following:

  $$\text{Fish}(x) \wedge \text{living}(x) \rightarrow \text{has\_scales}(x)$$

- This kind of notation is obviously more useful when writing logical expressions that are intended to be read by humans but when manipulated by a computer do not add any value.

Truth Tables

- We can use variables to represent possible truth values, in much the same way that variables are used in algebra to represent possible numerical values.
- We can then apply logical operators to these variables and can reason about the way in which they behave.
- It is usual to represent the behavior of these logical operators using truth tables.
- A truth table shows the possible values that can be generated by applying an operator to truth values.
- **Not**

- ➢ First of all, we will look at the truth table for not, $\neg$.
- ➢ Not is a unary operator, which means it is applied only to one variable.
- ➢ Its behavior is very simple:

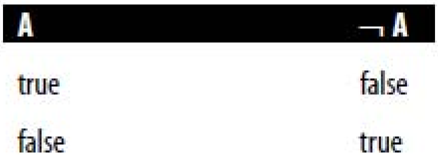
$\neg$ true is equal to false

$\neg$ false is equal to true

If variable A has value true, then $\neg$ A has value false.

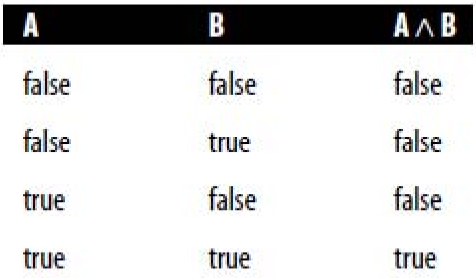
If variable B has value false, then $\neg$ B has value true.

- ➢ These can be represented by a truth table,



A	¬ A
true	false
false	true

- **And**
  - ➢ Now, let us examine the truth table for our first binary operator—one which acts on two variables:

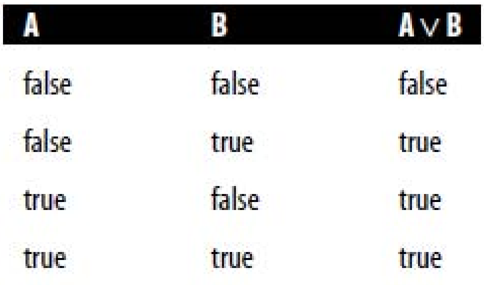


A	B	A ∧ B
false	false	false
false	true	false
true	false	false
true	true	true

- ➢ $\wedge$ is also called the conjunctive operator.
- ➢ A $\wedge$ B is the conjunction of A and B.
- ➢ You can see that the only entry in the truth table for which A $\wedge$ B is true is the one where A is true and B is true. If A is false, or if B is false, then A $\wedge$ B is false. If both A and B are false, then A $\wedge$ B is also false.
- ➢ What do A and B mean? They can represent any statement, or proposition, that can take on a truth value.

- For example, A might represent “It’s sunny,” and B might represent “It’s warm outside.” In this case, A ∧ B would mean “It is sunny and it’s warm outside,” which clearly is true only if the two component parts are true (i.e., if it is true that it is sunny and it is true that it is warm outside).

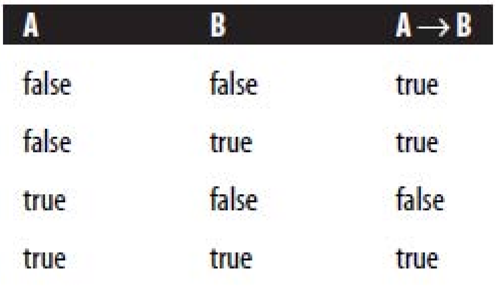
- **Or**
  - The truth table for the or operator, ∨



A	B	A ∨ B
false	false	false
false	true	true
true	false	true
true	true	true

  - ∨ is also called the disjunctive operator.
  - A ∨ B is the disjunction of A and B.
  - Clearly A ∨ B is true for any situation except when both A and B are false.
  - If A is true, or if B is true, or if both A and B are true, A ∨ B is true.
  - This table represents the inclusive-or operator.
  - A table to represent exclusive-or would have false in the final row. In other words, while A ∨ B is true if A and B are both true, A EOR B (A exclusive-or B) is false if A and B are both true.
  - You may also notice a pleasing symmetry between the truth tables for ∧ and ∨. This will become useful later, as will a number of other symmetrical relationships.

- **Implies**
  - The truth table for implies (→) is a little less intuitive.



A	B	A → B
false	false	true
false	true	true
true	false	false
true	true	true

  - This form of implication is also known as material implication
  - In the statement A→B, A is the antecedent, and B is the consequent. The bottom two lines of the table should be obvious. If A is true and B is true, then A →□B seems to be a reasonable thing to believe.
  - For example, if A means "you live in France" and B means "You speak French," then A→B corresponds to the statement "if you live in France, then you speak French."
  - Clearly, this statement is true (A→B is true) if I live in France and I speak French (A is true and B is true).
  - Similarly, if I live in France, but I don't speak French (A is true, but B is false), then it is clear that A→B is not true.
  - The situations where A is false are a little less clear. If I do not live in France (A is not true), then the truth table tells us that regardless of whether I speak French or not (the value of B), the statement A→B is true. A→B is usually read as "A implies B" but can also be read as "If A then B" or "If A is true then B is true."
  - Hence, if A is false, the statement is not really saying anything about the value of B, so B is free to take on any value (as long as it is true or false, of course!).
  - All of the following statements are valid:

    52 = 25 →4 = 4 (true →true)

    9 _ 9 = 123 →8 > 3 (false →true)

    52 = 25 →0 = 2 (false →false)

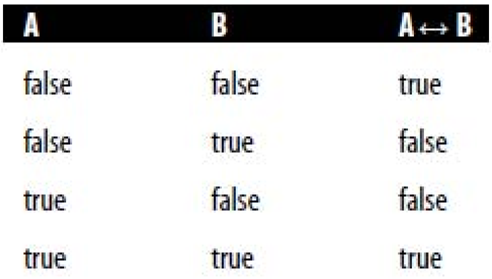
  - In fact, in the second and third examples, the consequent could be given any meaning, and the statement would still be true. For example, the following statement is valid:

    52 = 25 →Logic is weird

  - Notice that when looking at simple logical statements like these, there does not need to be any real-world relationship between the antecedent and the consequent.
  - For logic to be useful, though, we tend to want the relationships being expressed to be meaningful as well as being logically true.

- iff

- The truth table for iff (if and only if {↔}) is as follows:

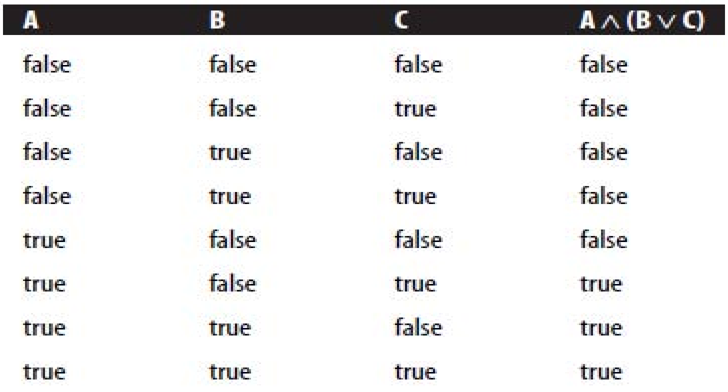


A	B	A↔B
false	false	true
false	true	false
true	false	false
true	true	true

- It can be seen that A ↔ B is true as long as A and B have the same value.
- In other words, if one is true and the other false, then A ↔ B is false. Otherwise, if A and B have the same value, A↔ B is true.

Complex Truth Tables

- Truth tables are not limited to showing the values for single operators.
- For example, a truth table can be used to display the possible values for A ∧(B ∨C).



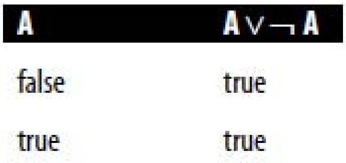
A	B	C	A ∧ (B ∨ C)
false	false	false	false
false	false	true	false
false	true	false	false
false	true	true	false
true	false	false	false
true	false	true	true
true	true	false	true
true	true	true	true

- Note that for two variables, the truth table has four lines, and for three variables, it has eight. In general, a truth table for n variables will have 2^n lines.
- The use of brackets in this expression is important. A ∧(B ∨C) is not the same as (A ∧B) ∨C.
- To avoid ambiguity, the logical operators are assigned precedence, as with mathematical operators.
- The order of precedence that is used is as follows: ¬ , ∧, ∨,→,↔

- Hence, in a statement such as $\neg A \lor \neg B \land C$, the $\neg$ operator has the greatest precedence, meaning that it is most closely tied to its symbols. $\land$ has a greater precedence than $\lor$, which means that the sentence above can be expressed as $(\neg A) \lor ((\neg B) \land C)$
- Similarly, when we write $\neg A \lor B$ this is the same as $(\neg A) \lor B$ rather than $\neg (A \lor B)$
- In general, it is a good idea to use brackets whenever an expression might otherwise be ambiguous.

Tautology

- Consider the following truth table:



A	$A \lor \neg A$
false	true
true	true

- This truth table has a property that we have not seen before: the value of the expression $A \lor \neg A$ is true regardless of the value of A.
- An expression like this that is always true is called a tautology.
- If A is a tautology, we write: $\models A$
- A logical expression that is a tautology is often described as being valid.
- A valid expression is defined as being one that is true under any interpretation.
- In other words, no matter what meanings and values we assign to the variables in a valid expression, it will still be true.
- For example, the following sentences are all valid:

  If wibble is true, then wibble is true.

  Either wibble is true, or wibble is not true.

- In the language of logic, we can replace wibble with the symbol A, in which case these two statements can be rewritten as

  $A \rightarrow A$

A ∨ ¬ A

- If an expression is false in any interpretation, it is described as being contradictory.
- The following expressions are contradictory:

A ∧ ¬ A

(A ∨ ¬ A)→(A ∧ ¬ A)

Equivalence

- Consider the following two expressions:

A ∧ B

B ∧ A

- It should be fairly clear that these two expressions will always have the same value for a given pair of values for A and B.
- In otherwords, we say that the first expression is logically equivalent to the second expression.
- We write this as A ∧ B _ B ∧ A. This means that the ∧ operator is commutative.
- Note that this is not the same as implication: A ∧ B→B ∧ A, although this second statement is also true.
- The difference is that if for two expressions e1 and e2: e1 _ e2, then e1 will always have the same value as e2 for a given set of variables.
- On the other hand, as we have seen, e1→e2 is true if e1 is false and e2 is true.
- There are a number of logical equivalences that are extremely useful.
- The following is a list of a few of the most common:

A ∨ A _ A

A ∧ A _ A

A ∧ (B ∧ C) _ (A ∧ B) ∧ C (∧ is associative)

A ∨ (B ∨ C) _ (A ∨ B) ∨ C (∨ is associative)

A ∧ (B ∨ C) _ (A ∧ B) ∨ (A ∧ C) (∧ is distributive over ∨)

A ∧ (A ∨ B) _ A

A ∨ (A ∧ B) _ A

A ∧ true _ A

A ∧ false _ false

A ∨ true _ true

A ∨ false _ A

- All of these equivalences can be proved by drawing up the truth tables for each side of the equivalence and seeing if the two tables are the same.
- The following is a very important equivalence: A→B _ ¬ A ∨ B
- We do not need to use the → symbol at all—we can replace it with a combination of ¬ and ∨ .
- Similarly, the following equivalences mean we do not need to use ∧ or↔:

A ∧ B _ ¬ (¬ A ∨ ¬ B)

A↔ B _ ¬ (¬ (¬ A ∨ B) ∨ ¬ (¬ B ∨ A))

- In fact, any binary logical operator can be expressed using ¬ and ∨ . This is a fact that is employed in electronic circuits, where nor gates, based on an operator called nor, are used. Nor is represented by ↓, and is defined as follows:

A ↓ B _ ¬ (A ∨ B)

- Finally, the following equivalences are known as DeMorgan's Laws:

A ∧ B _ ¬ (¬ A ∨ ¬ B)

A ∨ B _ ¬ (¬ A ∧ ¬ B)

- By using these and other equivalences, logical expressions can be simplified.
- For example, (C ∧ D) ∨ ((C ∧ D) ∧ E) can be simplified using the following rule: A ∨ (A ∧ B) _ A hence, (C ∧ D) ∨ ((C ∧ D) ∧ E) _ C ∧ D

- In this way, it is possible to eliminate subexpressions that do not contribute to the overall value of the expression.

Propositional Logic

- There are a number of possible systems of logic.
- The system we have been examining so far is called propositional logic.
- The language that is used to express propositional logic is called the propositional calculus.
- A logical system can be defined in terms of its syntax (the alphabet of symbols and how they can be combined), its semantics (what the symbols mean), and a set of rules of deduction that enable us to derive one expression from a set of other expressions and thus make arguments and proofs.
- **Syntax**
  - We have already examined the syntax of propositional calculus. The alphabet of symbols, _ is defined as follows

    $\sum$ = {true, false, ¬ ,→, (,), ∧ , ∨ ,↔, p1, p2, p3, . . . , pn, . . . }

  - Here we have used set notation to define the possible values that are contained within the alphabet $\sum$.
  - Note that we allow an infinite number of proposition letters, or propositional symbols, p1, p2, p3, . . . , and so on.
  - More usually, we will represent these by capital letters P, Q, R, and so on,
  - If we need to represent a very large number of them, we will use the subscript notation (e.g., p1).
  - An expression is referred to as a well-formed formula (often abbreviated as wff) or a sentence if it is constructed correctly, according to the rules of the syntax of propositional calculus, which are defined as follows.
  - In these rules, we use A, B, C to represent sentences. In other words, we define a sentence recursively, in terms of other sentences.
  - The following are wellformed sentences:

    P,Q,R. . .

    true, false

(A)

$\neg A$

$A \wedge B$

$A \vee B$

$A \rightarrow B$

$A \leftrightarrow B$

- Hence, we can see that the following is an example of a wff:

  $P \wedge Q \vee (B \wedge \neg C) \rightarrow A \wedge B \vee D \wedge (\neg E)$

- **Semantics**
  - The semantics of the operators of propositional calculus can be defined in terms of truth tables.
  - The meaning of $P \wedge Q$ is defined as "true when P is true and Q is also true."
  - The meaning of symbols such as P and Q is arbitrary and could be ignored altogether if we were reasoning about pure logic.
  - In other words, reasoning about sentences such as $P \vee Q \wedge \neg R$ is possible without considering what P, Q, and R mean.
  - Because we are using logic as a representational method for artificial intelligence, however, it is often the case that when using propositional logic, the meanings of these symbols are very important.
  - The beauty of this representation is that it is possible for a computer to reason about them in a very general way, without needing to know much about the real world.
  - In other words, if we tell a computer, "I like ice cream, and I like chocolate," it might represent this statement as $A \wedge B$, which it could then use to reason with, and, as we will see, it can use this to make deductions.

Predicate Calculus

- **Syntax**

- Predicate calculus allows us to reason about properties of objects and relationships between objects.
- In propositional calculus, we could express the English statement “I like cheese” by A. This enables us to create constructs such as $\neg$ A, which means “I do not like cheese,” but it does not allow us to extract any information about the cheese, or me, or other things that I like.
- In predicate calculus, we use predicates to express properties of objects. So the sentence “I like cheese” might be expressed as L(me, cheese) where L is a predicate that represents the idea of “liking.” Note that as well as expressing a property of me, this statement also expresses a relationship between me and cheese. This can be useful, as we will see, in describing environments for robots and other agents.
- For example, a simple agent may be concerned with the location of various blocks, and a statement about the world might be T(A,B), which could mean: Block A is on top of Block B.
- It is also possible to make more general statements using the predicate calculus.
- For example, to express the idea that everyone likes cheese, we might say $(\forall x)(P(x) \rightarrow L(x, C))$
- The symbol $\forall$ is read “for all,” so the statement above could be read as “for every x it is true that if property P holds for x, then the relationship L holds between x and C,” or in plainer English: “every x that is a person likes cheese.” (Here we are interpreting P(x) as meaning “x is a person” or, more precisely, “x has property P.”)
- Note that we have used brackets rather carefully in the statement above.
- This statement can also be written with fewer brackets: $\forall x\ P(x) \rightarrow L(x, C)$, $\forall$ is called the universal quantifier.
- The quantifier $\exists$ can be used to express the notion that some values do have a certain property, but not necessarily all of them: $(\exists x)(L(x,C))$
- This statement can be read “there exists an x such that x likes cheese.”
- This does not make any claims about the possible values of x, so x could be a person, or a dog, or an item of furniture. When we use the existential

quantifier in this way, we are simply saying that there is at least one value of x for which L(x,C) holds.

- ➢ The following is true: $(\forall x)(L(x,C)) \rightarrow (\exists x)(L(x,C))$, but the following is not: $(\exists x)(L(x,C)) \rightarrow (\forall x)(L(x,C))$

- Relationships between ∀ and ∃
  - ➢ It is also possible to combine the universal and existential quantifiers, such as in the following statement: $(\forall x)\ (\exists y)\ (L(x,y))$.
  - ➢ This statement can be read "for all x, there exists a y such that L holds for x and y," which we might interpret as "everyone likes something."
  - ➢ A useful relationship exists between ∀ and ∃. Consider the statement "not everyone likes cheese." We could write this as

    $\neg (\forall x)(P(x) \rightarrow L(x,C))$ -------------- (1)

  - ➢ As we have already seen, $A \rightarrow B$ is equivalent to $\neg A \vee B$. Using DeMorgan's laws, we can see that this is equivalent to $\neg (A \wedge \neg B)$. Hence, the statement (1) above, can be rewritten:

    $\neg (\forall x) \neg (P(x) \wedge \neg L(x,C))$ ------------- (2)

  - ➢ This can be read as "It is not true that for all x the following is not true: x is a person and x does not like cheese." If you examine this rather convoluted sentence carefully, you will see that it is in fact the same as "there exists an x such that x is a person and x does not like cheese." Hence we can rewrite it as $(\exists x)(P(x) \wedge \neg L(x,C))$ ------------- (3)
  - ➢ In making this transition from statement (2) to statement (3), we have utilized the following equivalence: $\exists x \cong \neg (\forall x) \neg$
  - ➢ In an expression of the form $(\forall x)(P(x, y))$, the variable x is said to be bound, whereas y is said to be free. This can be understood as meaning that the variable y could be replaced by any other variable because it is free, and the expression would still have the same meaning, whereas if the variable x were to be replaced by some other variable in P(x,y), then the meaning of the

expression would be changed: $(\forall x)(P(y, z))$ is not equivalent to $(\forall x)(P(x, y))$, whereas $(\forall x)(P(x, z))$ is.

- Note that a variable can occur both bound and free in an expression, as in $(\forall x)(P(x,y,z) \rightarrow (\exists y)(Q(y,z)))$
- In this expression, x is bound throughout, and z is free throughout; y is free in its first occurrence but is bound in $(\exists y)(Q(y,z))$. (Note that both occurrences of y are bound here.)
- Making this kind of change is known as substitution.
- Substitution is allowed of any free variable for another free variable.

- **Functions**
  - In much the same way that functions can be used in mathematics, we can express an object that relates to another object in a specific way using functions.
  - For example, to represent the statement "my mother likes cheese," we might use L(m(me),cheese)
  - Here the function m(x) means the mother of x. Functions can take more than one argument, and in general a function with n arguments is represented as $f(x1, x2, x3, \ldots, x_n)$

First-Order Predicate Logic

- The type of predicate calculus that we have been referring to is also called firstorder predicate logic (FOPL).
- A first-order logic is one in which the quantifiers $\forall$ and $\exists$ can be applied to objects or terms, but not to predicates or functions.
- So we can define the syntax of FOPL as follows. First,we define a term:
- A constant is a term.
- A variable is a term. f(x1, x2, x3, . . . , xn) is a term if x1, x2, x3, . . . , xn are all terms.
- Anything that does not meet the above description cannot be a term.
- For example, the following is not a term: $\forall x\ P(x)$. This kind of construction we call a sentence or a well-formed formula (wff), which is defined as follows.

- In these definitions, P is a predicate, x1, x2, x3, . . . , xn are terms, and A,B are wff 's. The following are the acceptable forms for wff 's:

  P(x1, x2, x3, . . . , xn)

  $\neg A$

  $A \wedge B$

  $A \vee B$

  $A \rightarrow B$

  $A \leftrightarrow B$

  $(\forall x)A$

  $(\exists x)A$

- An atomic formula is a wff of the form P(x1, x2, x3, . . . , xn).
- Higher order logics exist in which quantifiers can be applied to predicates and functions, and where the following expression is an example of a wff:

  $(\forall P)(\exists x)P(x)$

Soundness

- We have seen that a logical system such as propositional logic consists of a syntax, a semantics, and a set of rules of deduction.
- A logical system also has a set of fundamental truths, which are known as axioms.
- The axioms are the basic rules that are known to be true and from which all other theorems within the system can be proved.
- An axiom of propositional logic, for example, is $A \rightarrow (B \rightarrow A)$
- A theorem of a logical system is a statement that can be proved by applying the rules of deduction to the axioms in the system.
- If A is a theorem, then we write $\vdash A$
- A logical system is described as being sound if every theorem is logically valid, or a tautology.
- It can be proved by induction that both propositional logic and FOPL are sound.
- **Completeness**

  - A logical system is complete if every tautology is a theorem—in other words, if every valid statement in the logic can be proved by applying the rules of deduction to the axioms. Both propositional logic and FOPL are complete.

- **Decidability**
  - A logical system is decidable if it is possible to produce an algorithm that will determine whether any wff is a theorem. In other words, if a logical system is decidable, then a computer can be used to determine whether logical expressions in that system are valid or not.
  - We can prove that propositional logic is decidable by using the fact that it is complete.
  - We can prove that a wff A is a theorem by showing that it is a tautology. To show if a wff is a tautology, we simply need to draw up a truth table for that wff and show that all the lines have true as the result. This can clearly be done algorithmically because we know that a truth table for n values has 2n lines and is therefore finite, for a finite number of variables.
  - FOPL, on the other hand, is not decidable. This is due to the fact that it is not possible to develop an algorithm that will determine whether an arbitrary wff in FOPL is logically valid.
- **Monotonicity**
  - A logical system is described as being monotonic if a valid proof in the system cannot be made invalid by adding additional premises or assumptions.
  - In other words, if we find that we can prove a conclusion C by applying rules of deduction to a premise B with assumptions A, then adding additional assumptions A ¬ and B ¬ will not stop us from being able to deduce C.
  - Monotonicity of a logical system can be expressed as follows:

    If we can prove {A, B} ├ C,

    then we can also prove: {A, B, A_, B_} ├ C.
  - In other words, even adding contradictory assumptions does not stop us from making the proof in a monotonic system.
  - In fact, it turns out that adding contradictory assumptions allows us to prove anything, including invalid conclusions. This makes sense if we recall the line in

the truth table for →, which shows that false → true. By adding a contradictory assumption, we make our assumptions false and can thus prove any conclusion.

Modal Logics and Possible Worlds

- The forms of logic that we have dealt with so far deal with facts and properties of objects that are either true or false.
- In these classical logics, we do not consider the possibility that things change or that things might not always be as they are now.
- Modal logics are an extension of classical logic that allow us to reason about possibilities and certainties.
- In other words, using a modal logic, we can express ideas such as “although the sky is usually blue, it isn’t always” (for example, at night). In this way, we can reason about possible worlds.
- A possible world is a universe or scenario that could logically come about.
- The following statements may not be true in our world, but they are possible, in the sense that they are not illogical, and could be true in a possible world:

  Trees are all blue.

  Dogs can fly.

  People have no legs.

- It is possible that some of these statements will become true in the future, or even that they were true in the past.
- It is also possible to imagine an alternative universe in which these statements are true now.
- The following statements, on the other hand, cannot be true in any possible world:

  $A \wedge \neg A$

  $(x > y) \wedge (y > z) \wedge (z > x)$

- The first of these illustrates the law of the excluded middle, which simply states that a fact must be either true or false: it cannot be both true and false.
- It also cannot be the case that a fact is neither true nor false. This is a law of classical logic, it is possible to have a logical system without the law of the excluded middle, and in which a fact can be both true and false.

- The second statement cannot be true by the laws of mathematics. We are not interested in possible worlds in which the laws of logic and mathematics do not hold.
- A statement that may be true or false, depending on the situation, is called contingent.
- A statement that must always have the same truth value, regardless of which possible world we consider, is noncontingent.
- Hence, the following statements are contingent:

  $A \wedge B$

  $A \vee B$

  I like ice cream.

  The sky is blue.

- The following statements are noncontingent:

  $A \vee \neg A$

  $A \wedge \neg A$

  If you like all ice cream, then you like this ice cream.

- Clearly, a noncontingent statement can be either true or false, but the fact that it is noncontingent means it will always have that same truth value.
- If a statement A is contingent, then we say that A is possibly true, which is written ◊ A
- If A is noncontingent, then it is necessarily true, which is written □ A
- **Reasoning in Modal Logic**
  - It is not possible to draw up a truth table for the operators ◊ and □
  - The following rules are examples of the axioms that can be used to reason in this kind of modal logic:

    $\Box A \rightarrow \Diamond A$

    $\Box \neg A \rightarrow \neg \Diamond A$

    $\Diamond A \rightarrow \neg \Box A$

- Although truth tables cannot be drawn up to prove these rules, you should be able to reason about them using your understanding of the meaning of the ◊ and □ operators.

Possible world representations

- It describes method proposed by Nilsson which generalizes firtst order logic in the modeling of uncertain beliefs
- The method assigns truth values ranging from 0 to 1 to possible worlds
- Each set of possible worlds corresponds to a different interpretation of sentences contained in a knowledge base denoted as KB
- Consider the simple case where a KB contains only the single sentence S, S may be either true or false. We envision S as being true in one set of possible worlds W_1 and false in another set W_2 . The actual world , the one we are in, must be in one of the two sets, but we are uncertain which one. Uncertainty is expressed by assigning a probability P to W_1 and 1 – P to $W_{2.}$ We can say then that the probability of S being true is P
- When KB contains L sentences, $S_{1,...}$ S_L , more sets of possible worlds are required to represent all consistent truth value assignments. There are 2^L possible truth assignments for L sentences.
- Truth Value assignments for the set {P. P→Q, Q}

Consistent			Inconsistent		
P	Q	P → Q	P	Q	P → Q
True	True	True	True	True	False
True	False	False	True	False	True
False	True	True	False	True	False
False	False	True	False	False	False

- They are based on the use of the probability constraints

  $0 \le p_i \le 1$, and $\sum_i p_i = 1$