Introduction to Logic, Logic Programming and Languages

CSE 505 – Computing with Logic

Stony Brook University

http://www.cs.stonybrook.edu/~cse505

Overview

1. Introduction to Mathematical Formalizations in Logic
2. Propositional Logic or the logic of compound statements
3. Logical Arguments
4. Predicative Logic or the logic or quantified statements
5. Logic Programming (short basic introduction, applications, research at Stony Brook, groups)

A Puzzle

- Knights and Liars/Knaves: Knights always tell the truth; Liars/Knaves always lie.
  - Zoe: "Mel is a liar"
  - Mel: "Neither I nor Zoe are liars“
- Who's lying?

A Puzzle

- Knights and Liars/Knaves: Knights always tell the truth; Liars always lie.
  - Zoe: "Mel is a liar"
  - Mel: "Neither I nor Zoe are liars“

(1) $z \Leftrightarrow \sim m$

(2) $m \Leftrightarrow \sim(\sim m \vee \sim z)$

by logical equivalence:

$\sim(\sim m \vee \sim z) \equiv m \wedge z$

(2) becomes $m \Leftrightarrow m \wedge z$

z	m	(1)	(2)	(1) ∧ (2)
T	T	F	T	F
T	F	T	T	T
F	T	T	F	F
F	F	F	T	F

Mathematical Formalization

- Why formalize language?
  - to remove ambiguity
  - to represent facts on a computer and use it for **proving**, proof-checking, etc.

All people are mortal.
Socrates is a person. → *Socrates is mortal.*

$\forall x \; P(x) \rightarrow M(x)$
$P(S)$ → $P(S) \rightarrow M(S)$, $P(S)$ → $M(S)$

  - to detect **unsound** reasoning in arguments

I am lying.

Logic

- Mathematical logic is a tool for dealing with formal reasoning!
  - formalization of natural language and reasoning methods
- Logic does:
  - Assess if an **argument** is Valid or invalid
- Logic does not directly:
  - Assess the truth of atomic statements

Propositional Logic

- Or the ***logic of compound statements*** is the study of:
  - the structure (syntax) and
  - the meaning (semantics) of (simple and complex) propositions
- The key questions are:
  - ***How is the truth value of a complex proposition obtained from the truth value of its simpler components?***
  - ***Which propositions represent correct reasoning arguments?***

Propositional Logic

- A **proposition** is a sentence that is either **`true`** or **`false`**, but not both
- Examples of ***simple propositions***:
  - ***John is a student.***
  - 5+1 = 6
  - 426 > 1721
  - ***It is 82 degrees outside right now.***
- Example of a ***complex/composed*** proposition:
  - ***Tom is five and Mary is six.***
- Sentences which are not propositions:
  - ***Did Steve get an A on the exam?*** **(this is a *query*)**
  - ***Go away!*** **(this is an *order*)**

Propositional Logic

- In studying properties of propositions we represent them by expressions called **proposition forms** or ***formulas*** built from ***propositional variables*** (***atoms***), which represent simple propositions and symbols representing logical connectives
  - ***Proposition*** or ***propositional variables***: $p, q, \ldots$

    each can be true or false in 2-valued logics

    Examples: *p="Socrates is mortal."*

    q="Plato is mortal."
  - **Connectives**: $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\sim$
    - connect propositions: $p \vee q$
    - Example: "***I passed the exam or I did not pass it***." $p \vee \sim p$
      - The formula expresses the logical structure of the proposition, where p is an abbreviation for the simple proposition "I passed the exam."

Connectives

- ~ not
- $\wedge$ and
- $\vee$ or (non-exclusive!)
- $\rightarrow$ implies (if … then …)
- $\leftrightarrow$ if and only if
- $\forall$ for all
- $\exists$ exists

Formulas

- *Atomic*: $p, q, x, y, \ldots$
- *Unit Formula*: $p, \sim p$, (formula), $\ldots$
- Conjunctive: $p \land q, p \land \sim q, \ldots$
- Disjunctive: $p \lor q, p \lor (q \land x), \ldots$
- Conditional: $p \rightarrow q$
- Biconditional: $p \leftrightarrow q$

Negation (~ or ¬ or !)

- We use the symbol $\sim$ to denote negation
- Formalization (syntax): If p is a formula, then $\sim p$ is also a formula. We say that the second formula is the ***negation*** of the first
  - Examples: p, $\sim p$, and $\sim\sim p$ are all formulas
- Examples:
  - ***John went to the store yesterday*** (p).
  - ***John did not go to the store yesterday*** ($\sim p$).

Negation (~ or ¬ or !)

- Meaning (semantics):

  If a proposition is true, then its negation is false.

  If it is false, then its negation is true.

  - We express the connection semantics via a so-called ***truth table***:

Truth Table for $\sim p$

p	$\sim p$
T	F
F	T

Conjunction (∧ or & or •)

- We use the symbol ∧ to denote conjunction
- Syntax: If p and q are formulas, then $p \wedge q$ is also a formula.
- Semantics: If p is true and q is true, then $p \wedge q$ is true. In all other cases, $p \wedge q$ is false.

Truth Table for $p \wedge q$

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

Conjunction (∧ or & or •)

- Example:
  1. ***Bill went to the store.***
  2. ***Mary ate cantaloupe.***
  3. ***Bill went to the store and Mary ate cantaloupe.***
- If p and q abbreviate the first and second sentence, then the third is represented by the conjunction $p \land q$.

Inclusive Disjunction (∨ or | or +)

- We use the symbol ∨ to denote (inclusive) disjunction.
- Syntax: If p and q are formulas, then $p \vee q$ is also a formula.
- Semantics: If p is true or q is true or both are true, then $p \vee q$ is true. If p and q are both false, then $p \vee q$ is false.

Truth Table for $p \vee q$

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

Inclusive Disjunction (∨ or | or +)

- Example:
  - ***John works hard*** (p).
  - ***Mary is happy*** (q).
  - ***John works hard or Mary is happy*** ($p \lor q$).

Exclusive Disjunction ($\oplus$, XOR)

- We use the symbol $\oplus$ to denote exclusive disjunction.
- Syntax: If p and q are formulas, then $p \oplus q$ is also a formula.
- Semantics: An exclusive disjunction $p \oplus q$ is true if, and only if, one of p or q is true, but not both.

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

- Example:
  - ***Either John works hard or Mary is happy*** ($p \oplus q$)

Implication

- Example of proposition:

  If I do not pass the exam, then I will fail the course.

Corresponding formula: $\sim p \rightarrow q$

(More later …)

Determining Truth of A Formula

- Atomic formulae: given
- Compound formulae: via meaning of the connectives
  - The semantics of logical connectives determines how propositional formulas are evaluated depending on the truth values assigned to propositional variables
  - Each possible truth assignment or valuation for the propositional variables of a formula yields a truth value
    - The different possibilities can be summarized in a ***truth table***

Evaluation of formulas - Truth Tables

- A ***truth table*** for a formula lists all possible "situations" of truth or falsity, depending on the values assigned to the propositional variables of the formula

Truth Tables

- Example: If p, q and r are the propositions "***Peter [Quincy, Richard] will lend Sam money***," then Sam can deduce logically correct, that he will be able to borrow money whenever one of his three friends is willing to lend him some: $p \lor q \lor r$

p	q	r	$p \lor q \lor r$
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	T
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	F

- Each row in the truth table corresponds to one possible situation of assigning truth values to p, q and r

Truth Tables

- How many rows are there in a truth table with **`n`** propositional variables?
  - for **`n = 1`**, there are **two** rows, e.g.for ~ (negation)
  - for **`n = 2`**, there are **four** rows, e.g.:

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

  - for **`n = 3`**, there are **eight** rows, and so on.
- Do you see a pattern?

Constructing Truth Tables

- There are two choices (**`true`** or **`false`**) for each of **`n`** variables, so in general there are **`2*2*...*2 = 2`**^{n} rows for **`n`** variables
- A systematic procedure is necessary to make sure you construct all rows without duplicates
  - count in binary: **`000`**, **`001`**, **`010`**, **`011`**, **`100`**, . . .
- The rightmost column must be computed as a function of all the truth values in the row:

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

Constructing Truth Tables

- Example 1: $p \wedge \sim q$ (read "p and not q")

p	q	$\sim q$	$p \wedge \sim q$
T	T	F	F
T	F	T	T
F	T	F	F
F	F	T	F

- Note : it is usually necessary to evaluate all subformulas

Constructing Truth Tables

- Example 2: $p \wedge (q \vee r)$ (read "p and, in addition, q or r")

p	q	r	$q \vee r$	$p \wedge (q \vee r)$
T	T	T	T	T
T	T	F	T	T
T	F	T	T	T
T	F	F	F	F
F	T	T	T	F
F	T	F	T	F
F	F	T	T	F
F	F	F	F	F

- Note : it is usually necessary to evaluate all subformulas

Constructing Truth Tables

- Because it is clumsy and time-consuming to build large explicit truth tables, we will be interested in more efficient logical evaluation procedures.
  - Symbolic proofs with logical equivalences

    (See later) $\sim\sim p \equiv p$

p	$\sim p$	$\sim(\sim p)$
T	F	T
F	T	F

p and $\sim(\sim p)$ always have the same truth values, so they are logically equivalent

Language: Syntax of Formulas

- We backtrack a bit to formally define the syntax of logic
- The **formal language** of propositional logic can be specified by **grammar rules**
  - The **syntactic structure** of a complex logical expression (i.e., its parse tree) must be **unambiguous**

$$
\begin{aligned}
\langle\mathbf{proposition}\rangle \;::=\; & \langle\mathbf{variable}\rangle \\
\mid\; & (\sim\langle\mathbf{proposition}\rangle) \\
\mid\; & (\langle\mathbf{proposition}\rangle \land \langle\mathbf{proposition}\rangle) \\
\mid\; & (\langle\mathbf{proposition}\rangle \lor \langle\mathbf{proposition}\rangle) \\
& \ldots \\
\langle\mathbf{variable}\rangle \;::=\; & p \mid q \mid r \mid \ldots
\end{aligned}
$$

Ambiguities in Syntax of Formulas

- For example, the expression $p \land q \lor r$ can be interpreted in two different ways:

p	q	r	$p \land q$	$(p \land q) \lor r$	$q \lor r$	$p \land (q \lor r)$
F	F	T	F	**T**	T	**F**

- Parentheses are needed to avoid ambiguities
- The same problem arises in arithmetic: does 5+2 x 4 means (5+2) x 4 or 5+(2 x 4)?
  - The problem there is solved with priorities
- **Priorities in logic:** $\sim\ >\ \land\ >\ \lor\ >\ \to$
  - $\land$, $\lor$ and $\to$ **operators are** ***left associative***
  - ***$\sim$ is right associative***
- With $\land$ **ahead of $\lor$ in the precedence**, there is no ambiguity in $p \land q \lor r$

Precedence

- $\sim$ *highest*
- $\wedge$
- $\vee$
- $\rightarrow$, $\leftrightarrow$ *lowest*
- Avoid confusion - use '(' and ')':
  - $(p \wedge q) \vee x$

Logical Equivalence

- If two formulas evaluate to the same truth value in all situations, so that their truth tables are the same, they are said to be ***logically equivalent***
  - We write $p \equiv q$ to indicate that two formulas p and q are logically equivalent
    - If two formulas are logically equivalent, their syntax may be different, but their semantics is the same
    - The logical equivalence of two formulas can be established by inspecting the associated truth tables.
    - Note: Substituting logically inequivalent formulas is the source of most real-world reasoning errors

Logical Equivalence

- Disjunction is commutative:

p	q	$p \lor q$	$q \lor p$
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

Logical Equivalence

- Disjunction is associative:

p	q	r	$(p \lor q) \lor r$	$p \lor (q \lor r)$
T	T	T	T	T
T	T	F	T	T
T	F	T	T	T
T	F	F	T	T
F	T	T	T	T
F	T	F	T	T
F	F	T	T	T
F	F	F	F	F

- We will therefore ambiguously write $p \lor q \lor r$ to denote either $(p \lor q) \lor r$ or $p \lor (q \lor r)$. The ambiguity is usually of no consequence, as both formulas have the same meaning.

Logical Equivalence

- Is $\sim(p \land q)$ logically equivalent ($\equiv$) to $\sim p \land \sim q$?

p	q	$p \land q$	$\sim(p \land q)$	$\sim p$	$\sim q$	$\sim p \land \sim q$
T	T	T	F	F	F	F
T	**F**	**F**	**T**	**F**	**T**	**F**
F	**T**	**F**	**T**	**T**	**F**	**F**
F	F	F	T	T	T	T

- Lines 2 and 3 prove that this is **not the case**.

Logical Equivalence

- Is $\sim(p \wedge q)$ logically equivalent ($\equiv$) to $\sim p \vee \sim q$?

p	q	$p \wedge q$	$\sim(p \wedge q)$	$\sim p$	$\sim q$	$\sim p \vee \sim q$
T	T	T	**F**	F	F	**F**
T	F	F	**T**	F	T	**T**
F	T	F	**T**	T	F	**T**
F	F	F	**T**	T	T	**T**

- Yes.

De Morgan's Laws

- There are a number of important equivalences, including the following De Morgan's Laws:

$$\sim(p \land q) \equiv \sim p \lor \sim q$$

$$\sim(p \lor q) \equiv \sim p \land \sim q$$

  - These equivalences can be used to transform a formula into a logically equivalent one of a certain syntactic form, called a "normal form“
- Another useful logical equivalence is double negation:

$$\sim\sim p \equiv p$$

Using De Morgan's Laws

$\sim(\sim p \wedge \sim q) \equiv \sim\sim(p \vee q) \equiv p \vee q$

- The first equivalence is by De Morgan's Law, the second by double negation

- We have just derived a new equivalence: $p \vee q \equiv \sim(\sim p \wedge \sim q)$ (as equivalence can be used in both directions) which shows that disjunction can be expressed in terms of conjunction and negation!

Some Logical Equivalences

- You should be able to convince yourself of (i.e., prove) each of these:
  - Commutativity of $\wedge$: $p \wedge q \equiv q \wedge p$
  - Commutativity of $\vee$: $p \vee q \equiv q \vee p$
  - Associativity of $\wedge$: $p \wedge (q \wedge r) \equiv (p \wedge q) \wedge r$
  - Associativity of $\vee$: $p \vee (q \vee r) \equiv (p \vee q) \vee r$
  - Idempotence: $p \equiv p \wedge p \equiv p \vee p$
  - Absorption: $p \equiv p \wedge (p \vee q) \equiv p \vee (p \wedge q)$

Some Logical Equivalences

- Distributivity of $\wedge$: $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
- Distributivity of $\vee$: $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Contradictions: $p \wedge \mathrm{F} \equiv \mathrm{F} \equiv p \wedge \sim p$
- Identities: $p \wedge \mathrm{T} \equiv p \equiv p \vee \mathrm{F}$
- Tautologies: $p \vee \mathrm{T} \equiv \mathrm{T} \equiv p \vee \sim p$

Tautologies

- A ***tautology*** is a formula that is always true, no matter which truth values we assign to its variables.
- Consider the proposition "*I passed the exam or I did not pass the exam,*" the logical form of which is represented by the formula $p \vee \sim p$

p	$\sim p$	$p \vee \sim p$
T	F	**T**
F	T	**T**

  - This is a tautology, as we get T in every row of its truth table.

Contradictions

- A ***contradiction*** is a formula that is always false.
- The logical form of the proposition "I passed the exam and I did not pass the exam" is represented by $p \land \sim p$

p	$\sim p$	$p \land \sim p$
T	F	**F**
F	T	**F**

  - This is a contradiction, as we get F in every row of its truth table

Tautologies and contradictions

- Tautologies and contradictions are related

Theorem: If p is a tautology (contradiction) then $\sim p$ is a contradiction (tautology).

Example: $p \vee \sim p$ a tautology

Is $\sim(p \vee \sim p)$ a contradiction?

$\sim(p \vee \sim p) \equiv \sim p \wedge \sim\sim p \equiv \sim p \wedge p \equiv p \wedge \sim p$

Yes. because $\sim(p \vee \sim p) \equiv p \wedge \sim p$ and $p \wedge \sim p$ is a contradiction.

Implication ($\rightarrow$)

- Syntax: If p and q are formulas, then $p \rightarrow q$ (read "p implies q") is also a formula
- We call p the ***premise*** and q the ***conclusion*** of the implication.
- Semantics: If p is true and q is false, then $p \rightarrow q$ is false. In all other cases, $p \rightarrow q$ is true.

Truth Table for $p \rightarrow q$

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Implication ($\rightarrow$)

- Example:

  p: You get A's on all exams.

  q: You get an A in this course.

  $p \rightarrow q$: If you get A's on all exams, then you will get an A in this course.

Implication ($\rightarrow$)

- The semantics of implication is trickier than for the other connectives
  - if p and q are both true, clearly the implication $p \rightarrow q$ is true
  - if p is true but q is false, clearly the implication $p \rightarrow q$ is false
  - If the premise p is false **no conclusion can be drawn**, but both q being true and being false are consistent, so that the implication $p \rightarrow q$ is true in both cases
- Implication can also be expressed by other connectives, for example, $p \rightarrow q$ is logically equivalent to $\sim(p \land \sim q)$, which is equivalent with $\sim p \lor q$

Example: The Case of the Bad Defense Attorney

- Prosecutor:
  - *"If the defendant is guilty, then he had an accomplice."*
- Defense Attorney:
  - *"That's not true!!"*
- What did the defense attorney just claim?
  - $\sim(p \rightarrow q) \equiv \sim\sim(p \land \sim q) \equiv p \land \sim q$

  which means that "***the defendant is guilty and he did not have an accomplice***"

Biconditional

- Syntax: If p and q are formulas, then $p \leftrightarrow q$ (read "p if and only if (iff) q") is also a formula.
- Semantics: If p and q are either both true or both false, then $p \leftrightarrow q$ is true. Otherwise, $p \leftrightarrow q$ is false.

Truth Table for $p \leftrightarrow q$

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Biconditional

- Example:
  - p: Bill will get an A.
  - q: Bill studies hard.
  - $p \leftrightarrow q$: Bill will get an A **if and only if** Bill studies hard.
- The biconditional may be viewed as a shorthand for a conjunction of two implications, as $p \leftrightarrow q$ is logically equivalent to $(p \rightarrow q) \wedge (q \rightarrow p)$

Necessary and Sufficient Conditions

- The phrase "necessary and sufficient conditions" appears often in mathematics
- A proposition p is ***sufficient*** for q if $p \rightarrow q$ is a tautology.
  - Example: It is sufficient for a student to get A's in CSE114, CSE215, CSE214 in order to be admitted to become a CSE major
- A proposition p is ***necessary*** for q if q cannot be true without it: $\sim p \rightarrow \sim q$ (equivalent to $q \rightarrow p$).
  - Example: *It is necessary for a student to have a 3.0 GPA in the core courses to be admitted to become a CSE major.*

Necessary and Sufficient Conditions

Theorem: If a proposition p is both necessary and sufficient for q, then p and q are logically equivalent (and vice versa).

Tautologies and Logical Equivalence

Theorem: A propositional formula p is logically equivalent to q if and only if $p \leftrightarrow q$ is a tautology

- Proof:
  - (a) If $p \leftrightarrow q$ is a tautology, then p is logically equivalent to q

  Why? If $p \leftrightarrow q$ is a tautology, then it is true for all truth assignments. By the semantics of the biconditional, this means that p and q agree on every row of the truth table. Hence the two formulas are logically equivalent.

  - (b) If p is logically equivalent to q, then $p \leftrightarrow q$ is a tautology

  Why? If p and q logically equivalent, then they evaluate to the same truth value for each truth assignment. By the semantics of the biconditional, the formula $p \leftrightarrow q$ is true in all situations. ■

Related Implications

- **Implication:** $p \rightarrow q$
  - If you get A's on all exams, you get an A in the course.
- **Contrapositive:** $\sim q \rightarrow \sim p$
  - If you didn't get an A in the course, then you didn't get A's on all exams
- Note that implication is logically equivalent to the contrapositive

p	q	$p \rightarrow q$	$\sim q$	$\sim p$	$\sim q \rightarrow \sim p$
T	T	**T**	F	F	**T**
T	F	**F**	T	F	**F**
F	T	**T**	F	T	**T**
F	F	**T**	T	T	**T**

Related Implications

- **Converse:** $q \rightarrow p$
  - If you get an A in the course, then you got A's on all exams.
- **Inverse:** $\sim p \rightarrow \sim q$
  - If you didn't get A's on all exams, then you didn't get an A in the course.
- Note that converse is logically equivalent to the inverse

p	q	$q \rightarrow p$	$\sim p$	$\sim q$	$\sim p \rightarrow \sim q$
T	T	**T**	F	F	**T**
T	F	**T**	F	T	**T**
F	T	**F**	T	F	**F**
F	F	**T**	T	T	**T**

Deriving Logical Equivalences

- We can establish logical equivalence either via truth tables OR symbolically
- Example: $p \leftrightarrow q$ is logically equivalent to $(p \rightarrow q) \land (q \rightarrow p)$

p	q	$q \leftrightarrow p$		$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \land (q \rightarrow p)$
T	T	**T**		T	T	**T**
T	F	**F**		F	T	**F**
F	T	**F**		T	F	**F**
F	F	**T**		T	T	**T**

- Symbolic proofs are much like the simplifications you did in high school algebra - trial-and-error leads to experience and finally cunning

Symbolic proofs

- Example: $p \wedge q \equiv (p \vee \sim q) \wedge q$
  - Proof: which laws are used at each step?

$$
\begin{aligned}
(p \vee \sim q) \wedge q &\equiv q \wedge (p \vee \sim q) && (1)\\
&\equiv (q \wedge p) \vee (q \wedge \sim q) && (2)\\
&\equiv (q \wedge p) \vee \mathrm{F} && (3)\\
&\equiv (q \wedge p) && (4)\\
&\equiv p \wedge q && (5)
\end{aligned}
$$

Symbolic proofs

- Example: $p \land q \equiv (p \lor \sim q) \land q$
  - Proof: which laws are used at each step?

$$
\begin{array}{llll}
(p \lor \sim q) \land q & \equiv q \land (p \lor \sim q) & & \text{(1) Commutativity of } \land \\
 & \equiv (q \land p) \lor (q \land \sim q) & & \text{(2)} \\
 & \equiv (q \land p) \lor \mathrm{F} & & \text{(3)} \\
 & \equiv (q \land p) & & \text{(4)} \\
 & \equiv p \land q & & \text{(5)}
\end{array}
$$

Symbolic proofs

- Example: $p \land q \equiv (p \lor \sim q) \land q$
  - Proof: which laws are used at each step?

$(p \lor \sim q) \land q$	$\equiv q \land (p \lor \sim q)$	(1) Commutativity of $\land$
	$\equiv (q \land p) \lor (q \land \sim q)$	(2) Distributivity of $\land$
	$\equiv (q \land p) \lor \mathrm{F}$	(3)
	$\equiv (q \land p)$	(4)
	$\equiv p \land q$	(5)

Symbolic proofs

- Example: $p \land q \equiv (p \lor \sim q) \land q$
  - Proof: which laws are used at each step?

$(p \lor \sim q) \land q$	$\equiv q \land (p \lor \sim q)$	(1) Commutativity of $\land$
	$\equiv (q \land p) \lor (q \land \sim q)$	(2) Distributivity of $\land$
	$\equiv (q \land p) \lor \mathrm{F}$	(3) Contradiction
	$\equiv (q \land p)$	(4)
	$\equiv p \land q$	(5)

Symbolic proofs

- Example: $p \wedge q \wedge r \equiv (p \vee \sim q) \wedge q$
  - Proof: which laws are used at each step?

$(p \vee \sim q) \wedge q$	$\equiv q \wedge (p \vee \sim q)$	(1) Commutativity of $\wedge$
	$\equiv (q \wedge p) \vee (q \wedge \sim q)$	(2) Distributivity of $\wedge$
	$\equiv (q \wedge p) \vee \mathrm{F}$	(3) Contradiction
	$\equiv (q \wedge p)$	(4) Identity
	$\equiv p \wedge q$	(5)

Symbolic proofs

- Example: $p \land q \equiv (p \lor \sim q) \land q$
  - Proof: which laws are used at each step?

$(p \lor \sim q) \land q$	$\equiv q \land (p \lor \sim q)$	(1) Commutativity of $\land$
	$\equiv (q \land p) \lor (q \land \sim q)$	(2) Distributivity of $\land$
	$\equiv (q \land p) \lor \mathrm{F}$	(3) Contradiction
	$\equiv (q \land p)$	(4) Identity
	$\equiv p \land q$	(5) Commutativity of $\land$

Logical Consequence

- We say that p logically implies q, or that q is a ***logical consequence*** of p, if q is true whenever p is true.
- Example: p logically implies $p \vee q$

p	q	$p \vee q$
T	T	**T**
T	F	**T**
F	T	T
F	F	F

- Note that logical consequence is a weaker condition than logical equivalence

Logical Arguments

- An *argument* (or *argument form*) is a (finite) sequence of statements (forms), usually written as follows:

$$p_1$$
$$\ldots$$
$$p_n$$
$$\therefore q$$

- We call $p_1, \ldots, p_n$ the premises (or assumptions or hypotheses) and q the conclusion, of the argument.
- We read:

"p_1, p_2, …, p_n, therefore q" OR

"From premises p_1, p_2, …, p_n infer conclusion q"

- Argument forms are also called *inference rules.*

Logical Arguments

- An inference rule is said to be *valid*, or (*logically*) *sound*, if it is the case that, for each truth valuation, if all the premises true, then the conclusion is also true!

Theorem: An inference rule is valid if, and only if, the conditional $p_1 \wedge p_2 \wedge \ldots \wedge p_n \rightarrow q$ is a tautology.

- An argument form consisting of two premises and a conclusion is called a *syllogism*.

Determining Validity or Invalidity

- **Testing an Argument Form for Validity**
  1. Identify the premises and conclusion of the argument form.
  2. Construct a truth table showing the truth values of all the premises and the conclusion.
  3. A row of the truth table in which **all the premises are true** is called a **critical row.** If there is a critical row in which the conclusion is false, then the argument form is invalid. If the conclusion in every critical row is true, then the argument form is valid

Determining Validity or Invalidity

$p \rightarrow q \vee \sim r$

$q \rightarrow p \wedge r$

$\therefore p \rightarrow r$

						premises		conclusion
p	q	r	$\sim r$	$q \vee \sim r$	$p \wedge r$	$p \rightarrow q \vee \sim r$	$q \rightarrow p \wedge r$	$p \rightarrow r$
T	T	T	F	T	T	T	T	T
T	T	F	T	T	F	T	F	F
T	F	T	F	F	T	F	T	F
T	F	F	T	T	F	T	T	F
F	T	T	F	T	F	T	F	F
F	T	F	T	T	F	T	F	F
F	F	T	F	F	F	T	T	T
F	F	F	T	T	F	T	T	T

This row shows it is possible for an argument of this form to have true premises and a false conclusion. Hence this form of argument is invalid.

Invalid argument

Modus Ponens

- Modus Ponens: $p \rightarrow q$

  "method of affirming" p

  Latin $\therefore q$

p	q	$p \to q$	p	q
		premises		conclusion
T	T	T	T	T ← critical row
T	F	F	T	F
F	T	T	F	T
F	F	T	F	F

Valid argument

Modus Ponens

- The following argument is valid:

  If Socrates is a man, then Socrates is mortal.

  Socrates is a man.

  ∴ Socrates is mortal.

Modus Tollens

- Modus Tonens: $p \rightarrow q$

"method of denying" $\sim q$

Latin $\therefore \sim p$

- Modus Tollens is valid because :
  - modus ponens is valid and the fact that a conditional statement is logically equivalent to its contrapositive, OR
  - it can be established formally by using a truth table.

Modus Tollens

- Example:

  (1) If Zeus is human, then Zeus is mortal.

  (2) Zeus is not mortal.

  ∴ Zeus is not human.

- An intuitive proof is proof by contradiction
  - if Zeus were human, then by (1) he would be mortal.
  - But by (2) he is not mortal.
  - Hence, Zeus cannot be human.

Recognizing Modus Ponens and Modus Tollens

If there are more pigeons than there are pigeonholes, then at least two pigeons roost in the same hole.

There are more pigeons than there are pigeonholes.

∴ ?

Recognizing Modus Ponens and Modus Tollens

If there are more pigeons than there are pigeonholes, then at least two pigeons roost in the same hole.

There are more pigeons than there are pigeonholes.

∴ At least two pigeons roost in the same hole.

by modus ponens

Recognizing Modus Ponens and Modus Tollens

If 870,232 is divisible by 6, then it is divisible by 3.

870,232 is **not** divisible by 3.

$\therefore$?

Recognizing Modus Ponens and Modus Tollens

If 870,232 is divisible by 6, then it is divisible by 3.

870,232 is **not** divisible by 3.

∴ 870,232 is not divisible by 6.

by modus tollens

Other Rules of Inference

- **Generalization:**

$$p \qquad\qquad \textbf{and} \qquad\qquad q$$

$$\therefore p \lor q \qquad\qquad\qquad\qquad \therefore p \lor q$$

- Example:

Anton is a junior.

∴ (more generally) Anton is a junior or Anton is a senior.

Other Rules of Inference

- **Specialization:**

$$p \wedge q \qquad \textbf{and} \qquad p \wedge q$$
$$\therefore p \qquad\qquad\qquad \therefore q$$

- Example:

  Ana knows numerical analysis and

  Ana knows graph algorithms.

  ∴ (in particular) Ana knows graph algorithms.

Other Rules of Inference

- **Elimination :**

$p \lor q$ **and** $p \lor q$

$\sim q$ $\sim p$

$\therefore p$ $\therefore q$

  - If we have only two possibilities and we can rule one out, the other one must be the case
- Example:

$x - 3 = 0$ or $x + 2 = 0$

$x + 2 \neq 0.$

$\therefore x - 3 = 0.$

Other Rules of Inference

- **Transitivity :**

$$p \longrightarrow q$$
$$q \longrightarrow r$$
$$\therefore p \longrightarrow r$$

- Example:

If 18,486 is divisible by 18, then 18,486 is divisible by 9.

If 18,486 is divisible by 9, then the sum of the digits of 18,486 is divisible by 9.

∴ If 18,486 is divisible by 18, then the sum of the digits of 18,486 is divisible by 9.

Proof Techniques

- **Proof by Contradiction:**

$$\sim p \rightarrow c, \textit{ where } c \textit{ is a contradiction}$$

$$\therefore p$$

  - The usual way to derive a conditional $\sim p \rightarrow c$ is to assume $\sim p$ and then derive c (i.e., a contradiction).
  - Thus, if one can derive a contradiction from $\sim p$, then one may conclude that p is true.

The Logic of Quantified Statements

All men are mortal.

Socrates is a man.

∴ Socrates is mortal.

- *Propositional calculus*: analysis of ordinary compound statements
- ***Predicate calculus*** or ***The Logic of Quantified Statements***: symbolic analysis of predicates and **quantified statements (∀x, ∃x)**
  - *Example: P* is a *predicate symbol*

    P stands for "*is a student at SBU*"

    P(x) stands for "*x is a student at SBU*"
  - *x* is a *predicate variable*

The Logic of Quantified Statements

- A ***predicate*** is a sentence that contains a finite number of variables and becomes a ***statement*** (or ***ground predicate***) when specific values are substituted for the variables.
- The ***domain*** of a predicate variable is the set of all values that may be substituted in place of the variable.
  - Example:

$P(x)$ is the predicate "$x^2 > x$", x has as a domain the set **R** of all real numbers

$P(2)$: $2^2 > 2$. *True.*

$P(1/2)$: $(1/2)^2 > 1/2$. *False.*

Truth Set of a Predicate

- If $P(x)$ is a predicate and x has domain D, the truth set of $P(x)$, the *truth set of P*, $\{x \in D \mid P(x)\}$, is the set of all elements of D that make $P(x)$ true when they are substituted for x.
  - Example:

    $Q(n)$ is the predicate for "*n* is a factor of *8*."

    if the domain of n is the set **Z** of all integers

    The truth set is $\{1, 2, 4, 8, -1, -2, -4, -8\}$

The Universal Quantifier: $\forall$

- Quantifiers are words that refer to quantities ("*some*" or "*all*") and tell for how many elements a given predicate is true.
- **Universal quantifier:** $\forall$ "for all"
  - Example:

    $\forall$ human beings x, x is mortal.

    "All human beings are mortal"

    - If H is the set of all human beings:

    $\forall x \in H$, x is mortal

Universal statements

- A **universal statement** is a statement of the form

"$\forall x \in D, Q(x)$" where $Q(x)$ is a predicate and D is the domain of x.

  - $\forall x \in D, Q(x)$ is true if, and only if, $Q(x)$ is true for every x in D
  - $\forall x \in D, Q(x)$ is false if, and only if, $Q(x)$ is false for at least one x in D (the value for x is a **counterexample**)

- Example:

$\forall x \in D, x^2 \geq x$ for $D = \{1, 2, 3, 4, 5\}$

$1^2 \geq 1,\quad 2^2 \geq 2,\quad 3^2 \geq 3,\quad 4^2 \geq 4,\quad 5^2 \geq 5$

  - Hence "$\forall x \in D, x^2 \geq x$" is true.

The Existential Quantifier: ∃

- **Existential quantifier:** ∃ "there exists"
- Example:
  - "There is a student in the course"

∃ a person p such that p is a student in the course

$\exists p \in P$ such that p is a student in the course

where P is the set of all people

The Existential Quantifier: ∃

- An **existential statement** is a statement of the form

"$\exists x \in D$ such that $Q(x)$" where $Q(x)$ is a predicate and D the domain of x

  - $\exists x \in D$ s.t. $Q(x)$ is true if, and only if, $Q(x)$ is true for at least one x in D
  - $\exists x \in D$ s.t. $Q(x)$ is false if, and only if, $Q(x)$ is false for all x in D
- Example:
  - $\exists m \in \boldsymbol{Z}$ such that $m^2 = m$

    It is true. Example: $1^2 = 1$

Notation: such that = s.t.

Universal Conditional Statements

- **Universal conditional statement:**

$\forall x$, *if* $P(x)$ *then* $Q(x)$

- Example:

  If a real number is greater than 2 then its square is greater than 4.

$$\forall x \in \boldsymbol{R}, \text{ if } x > 2 \text{ then } x^2 > 4$$

Equivalent Forms of Universal and Existential Statements

- $\forall x \in U$, *if* $P(x)$ *then* $Q(x)$ can be rewritten in the form $\forall x \in D$, $Q(x)$ by narrowing U to be the domain D consisting of all values of the variable x that make $P(x)$ true.
  - Example: $\forall x$, *if x is a square then x is a rectangle*
    $\forall$ *squares x, x is a rectangle.*
- $\exists x$ *such that* $P(x)$ *and* $Q(x)$ can be rewritten in the form $\exists x \in D$ *such that* $Q(x)$ where D consists of all values of the variable x that make $P(x)$ true

Implicit Quantification

- $P(x) \Rightarrow Q(x)$ means that every element in the truth set of $P(x)$ is in the truth set of $Q(x)$, or, equivalently, $\forall x, P(x) \rightarrow Q(x)$
- $P(x) \Leftrightarrow Q(x)$ means that $P(x)$ and $Q(x)$ have identical truth sets, or, equivalently, $\forall x$, $P(x) \leftrightarrow Q(x)$.

Negations of Quantified Statements

- Negation of a Universal Statement:

The negation of a statement of the form $\forall x \in D, Q(x)$ is logically equivalent to a statement of the form $\exists x \in D, \sim Q(x)$:

$$\sim(\forall x \in D, Q(x)) \equiv \exists x \in D, \sim Q(x)$$

- Example:
  - "All mathematicians wear glasses"
  - Its negation is: "There is at least one mathematician who does not wear glasses"
  - Its negation is NOT "No mathematicians wear glasses"

Negations of Quantified Statements

- Negation of an Existential Statement

The negation of a statement of the form $\exists x \in D, Q(x)$

is logically equivalent to a statement of the form $\forall x \in D, \sim Q(x)$:

$$\sim(\exists x \in D, Q(x)) \equiv \forall x \in D, \sim Q(x)$$

- Example:
  - "Some snowflakes are the same."
  - Its negation is: "All snowflakes are different."

Negations of Quantified Statements

- More Examples:
  - $\sim(\forall$ primes p, p is odd) $\equiv \exists$ a prime p such that p is **not** odd
  - $\sim(\exists$ a triangle T such that the sum of the angles of T equals $200°$) $\equiv$ $\forall$ triangles T, the sum of the angles of T **does not** equal $200°$
  - $\sim(\forall$ politicians x, x is **not** honest) $\equiv \exists$ a politician x such that x is honest **(by double negation)**
  - $\sim(\forall$ computer programs p, p is finite) $\equiv \exists$ a computer program p that is not finite
  - $\sim(\exists$ a computer hacker c, c is over 40) $\equiv \forall$ computer hacker c, c is 40 or under
  - $\sim(\exists$ an integer n between 1 and 37 such that 1,357 is divisible by n) $\equiv \forall$ integers n between 1 and 37, 1,357 is not divisible by n

Negations of Universal Conditional Statements

- $\sim(\forall x,\ P(x) \rightarrow Q(x)) \ \equiv\ \exists x \text{ such that } P(x) \wedge \sim Q(x)$

Proof:

$$\sim(\forall x,\ P(x) \rightarrow Q(x)) \equiv \exists x \text{ such that } \sim(P(x) \rightarrow Q(x))$$

and

$$\sim(P(x) \rightarrow Q(x)) \equiv \sim(\sim P(x) \vee Q(x)) \equiv \sim\sim P(x) \wedge \sim Q(x)$$
$$\equiv P(x) \wedge \sim Q(x)$$

- Examples:
  - $\sim$($\forall$ people p, if p is blond then p has blue eyes) $\equiv$
    $\exists$ a person p such that p is blond and p does not have blue eyes
  - $\sim$(If a computer program has more than 100,000 lines, then it contains a bug) $\equiv$ There is at least one computer program that has more than 100,000 lines and does not contain a bug

The Relation among ∀, ∃, ∧, and ∨

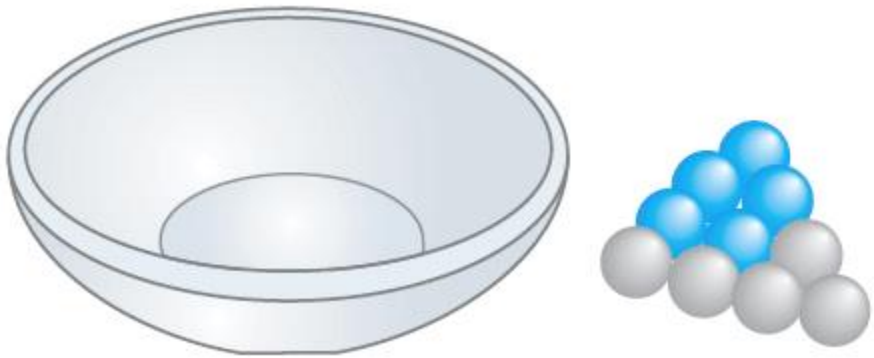
- $D = \{x_1, x_2, \ldots, x_n\}$ and $\forall x \in D, Q(x)$

  $\equiv Q(x_1) \wedge Q(x_2) \wedge \cdots \wedge Q(x_n)$

- $D = \{x_1, x_2, \ldots, x_n\}$ and $\exists x \in D$ such that $Q(x)$

  $\equiv Q(x_1) \vee Q(x_2) \vee \cdots \vee Q(x_n)$

Vacuous Truth of Universal Statements



All the balls in the bowl are blue?

True

$\forall x$ in D, if $P(x)$ then $Q(x)$ is ***vacuously true*** or ***true by default*** if, and only if, $P(x)$ is false for every x in D

Variants of Universal Conditional Statements

- Universal conditional statement: $\forall x \in D$, if $P(x)$ then $Q(x)$
- **Contrapositive**: $\forall x \in D$, if $\sim Q(x)$ then $\sim P(x)$

  $\forall x \in D$, if $P(x)$ then $Q(x) \equiv \forall x \in D$, if $\sim Q(x)$ then $\sim P(x)$

  Proof: for any x in D by the logical equivalence between statement and its contrapositive
- **Converse:** $\forall x \in D$, if $Q(x)$ then $P(x)$.
- **Inverse:** $\forall x \in D$, if $\sim P(x)$ then $\sim Q(x)$.
- Example:

  $\forall x \in R$, if $x > 2$ then $x^2 > 4$

  Contrapositive: $\forall x \in R$, if $x^2 \leq 4$ then $x \leq 2$

  Converse: $\forall x \in R$, if $x^2 > 4$ then $x > 2$

  Inverse: $\forall x \in R$, if $x \leq 2$ then $x^2 \leq 4$

Necessary and Sufficient Conditions

- Necessary condition:

"$\forall x$, r (x) is a **necessary condition** for s(x)" means

"$\forall x$, if ~r (x) then ~s(x)" $\equiv$ "$\forall x$, if s(x) then r(x)" (*)

(*)(by contrapositive and double negation)

- Sufficient condition:

"$\forall x$, r (x) is a **sufficient condition** for s(x)" means

"$\forall x$, if r (x) then s(x)"

Necessary and Sufficient Conditions

- Examples:
  - Squareness is a **sufficient condition** for rectangularity;

Formal statement:

$\forall$x, if x is a square, then x is a rectangle

  - Being at least 35 years old is a **necessary condition** for being President of the United States

$\forall$ people x, if x is younger than 35, then x cannot be President of the United States $\equiv$

$\forall$ people x, if x is President of the United States then x is at least 35 years old (by contrapositive)

Statements with Multiple Quantifiers

- Example:

"*There is a person supervising every detail of the production process*"

  - What is the meaning?

"***There is one single person who supervises all the details of the production process***"?

OR

"***For any particular production detail, there is a person who supervises that detail, but there might be different supervisors for different details***"?

NATURAL LANGUAGE IS AMBIGUOUS

LOGIC IS CLEAR

Statements with Multiple Quantifiers

- In Logic: Quantifiers are performed in the order in which the quantifiers occur:
  - Examples:

$\forall$x in set D, $\exists$y in set E such that x and y satisfy property P(x, y)

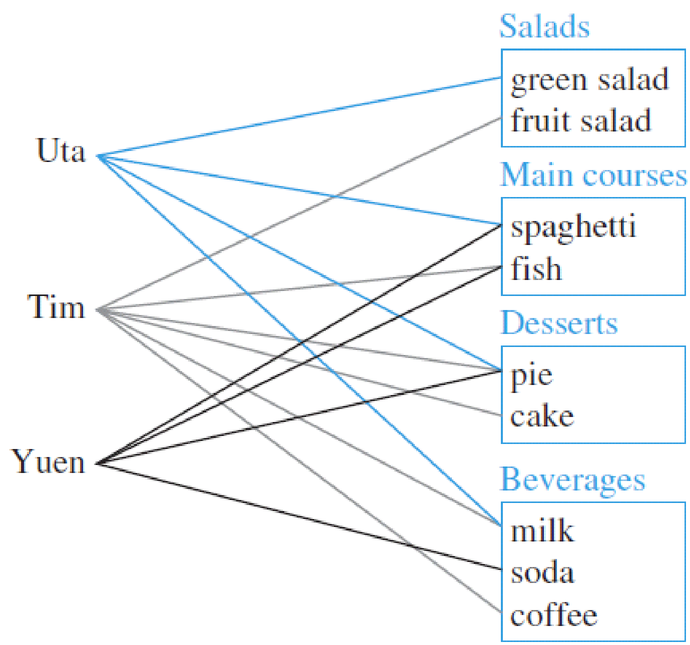
is different from:

$\exists$y in set E such that $\forall$x in set D, x and y satisfy property P(x, y)

Interpreting Statements with Two Different Quantifiers

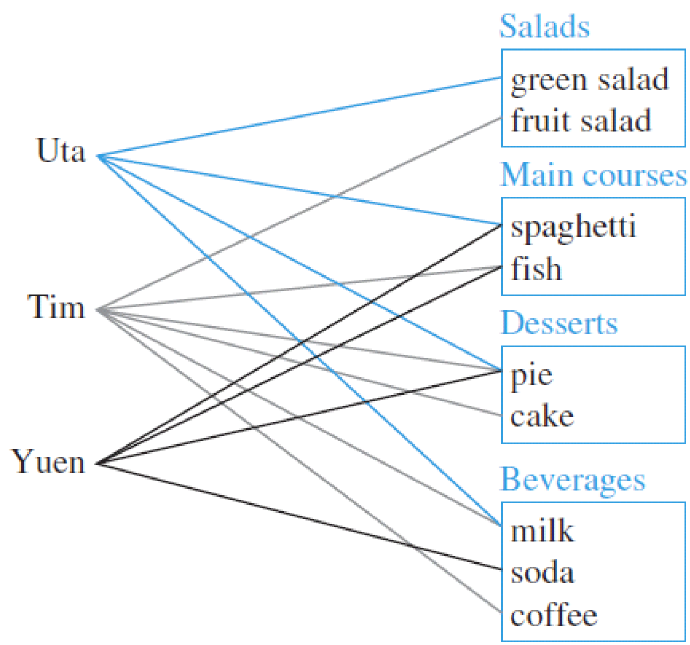
- Explanations:
- $\forall$x in D, $\exists$y in E such that P(x, y)
  - for whatever element x in D you must find an element y in E that "works" for that particular x
- $\exists$y in E such that $\forall$x in D, P(x, y)
  - find one particular y in E that will "work" no matter what x in D anyone might choose

Interpreting Statements with Two Different Quantifiers



- ∃ an item I such that ∀ students S, S chose I .
- ∃ a student S such that ∀ stations Z, ∃ an item I in Z such that S chose I
- ∀ students S and ∀ stations Z, ∃ an item I in Z such that S chose I .

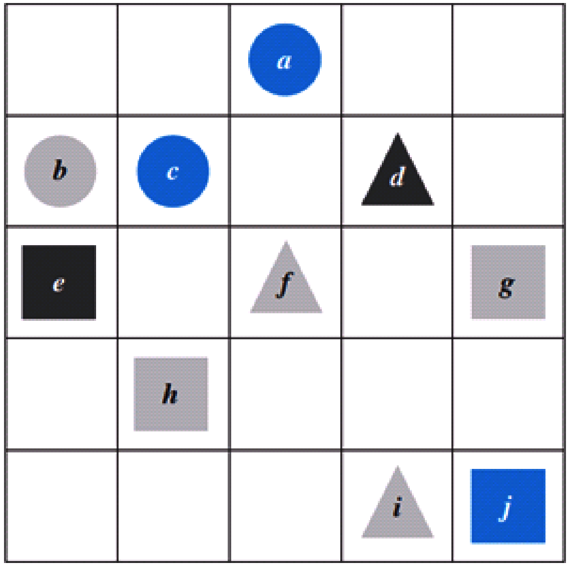
Interpreting Statements with Two Different Quantifiers



- ∃ an item I such that ∀ students S, S chose I . TRUE
- ∃ a student S such that ∀ stations Z, ∃ an item I in Z such that S chose I TRUE
- ∀ students S and ∀ stations Z, ∃ an item I in Z such that S chose I . FALSE

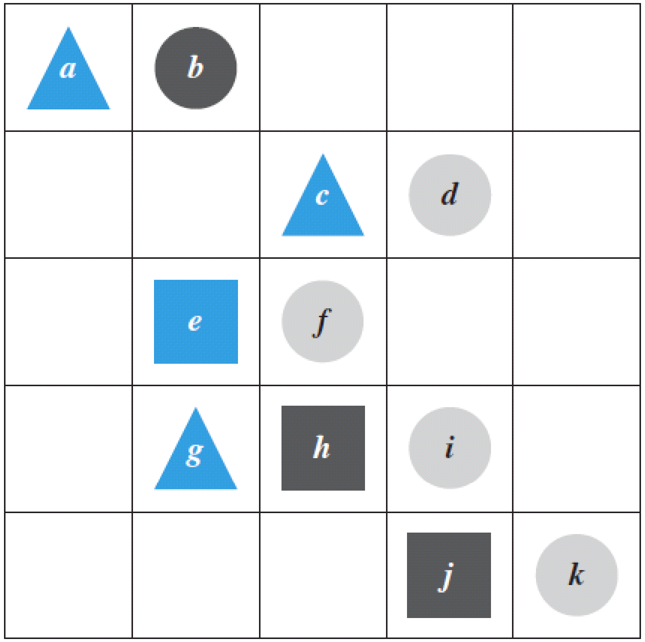
Tarski's World is a good world to Formalizing Logic Statements

- Blocks of various sizes, shapes, and colors located on a grid
  - Triangle(x) means "x is a triangle"
  - Circle(x) means "x is a circle"
  - Square(x) means "x is a square"
  - Blue(x) means "x is blue"
  - Gray(x) means "x is gray"
  - Black(x) means "x is black"



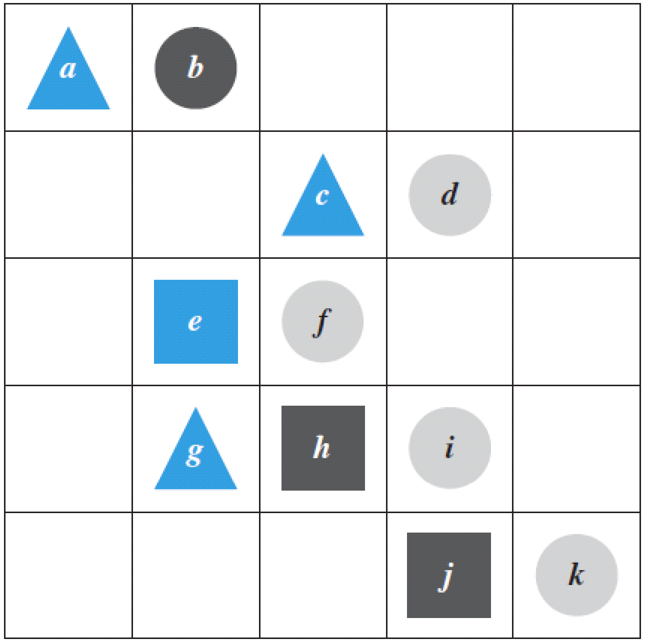
  - RightOf(x, y) means "x is to the right of y"
  - Above(x, y) means "x is above y"
  - SameColorAs(x, y) means "x has the same color as y"
  - x = y denotes the predicate "x is equal/same to y"

Tarski's World



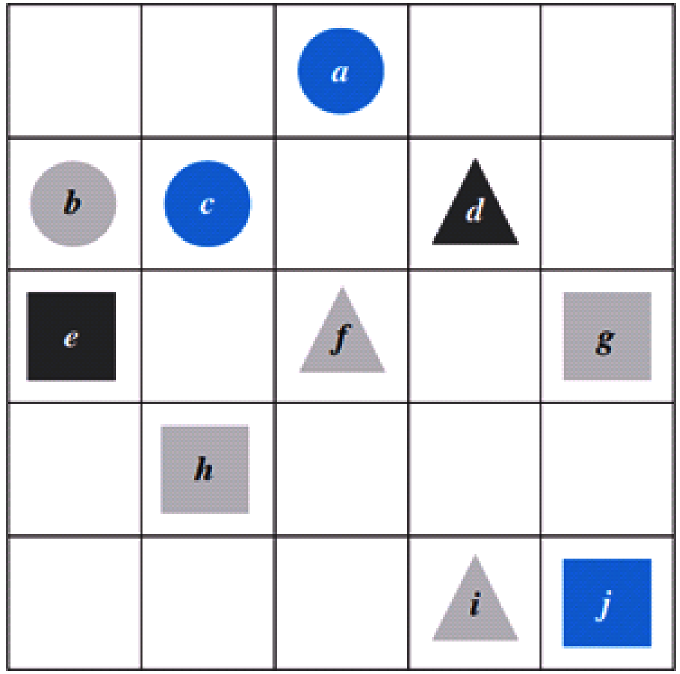
- $\forall t$, Triangle(t) $\rightarrow$ Blue(t). TRUE
- $\forall x$, Blue(x) $\rightarrow$ Triangle(x). FALSE
- $\exists y$ such that Square(y) $\wedge$ RightOf(d, y). TRUE
- $\exists z$ such that Square(z) $\wedge$ Gray(z). FALSE

Tarski's World



- $\forall t$, Triangle(t) $\rightarrow$ Blue(t).
- $\forall x$, Blue(x) $\rightarrow$ Triangle(x).
- $\exists y$ such that Square(y) $\wedge$ RightOf(d, y).
- $\exists z$ such that Square(z) $\wedge$ Gray(z).

Statements with Multiple Quantifiers in Tarski's World



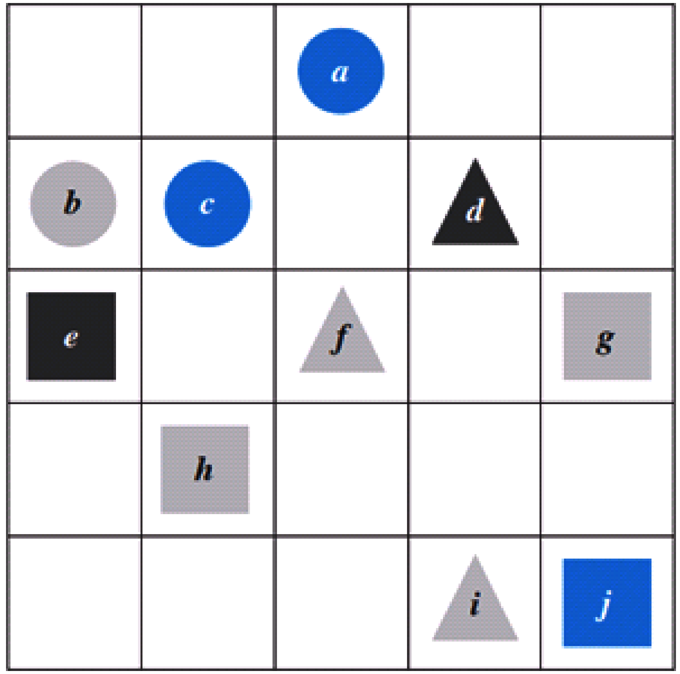
$\forall\exists$

- For all triangles x, there is a square y such that x and y have the same color

TRUE

Given $x =$	**choose** $y =$	**and check that** y **is the same color as** x**.**
d	e	yes ✓
f or i	h or g	yes ✓

Statements with Multiple Quantifiers in Tarski's World



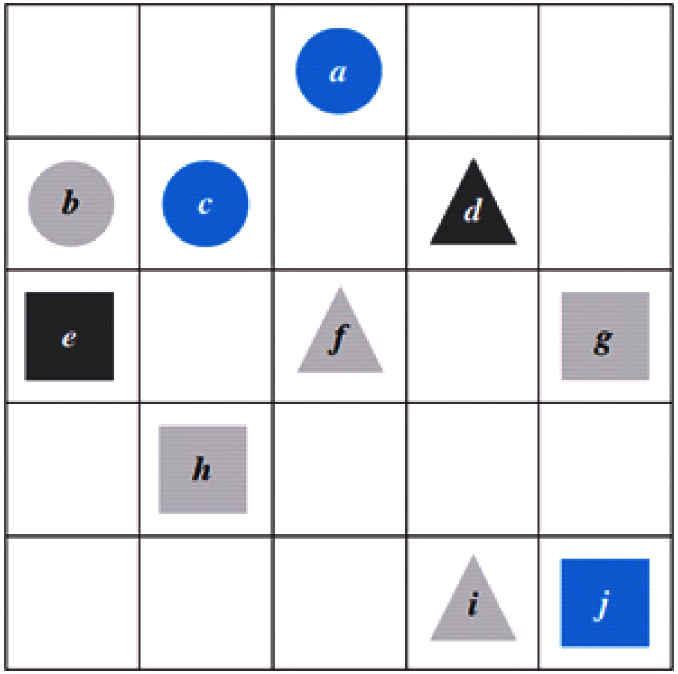
$\exists\forall$

- There is a square y such that, for all triangles x, x and y have the same color

FALSE

there is no such square

Quantifier Order in Tarski's World



- For every square x there is a triangle y such that x and y have different colors

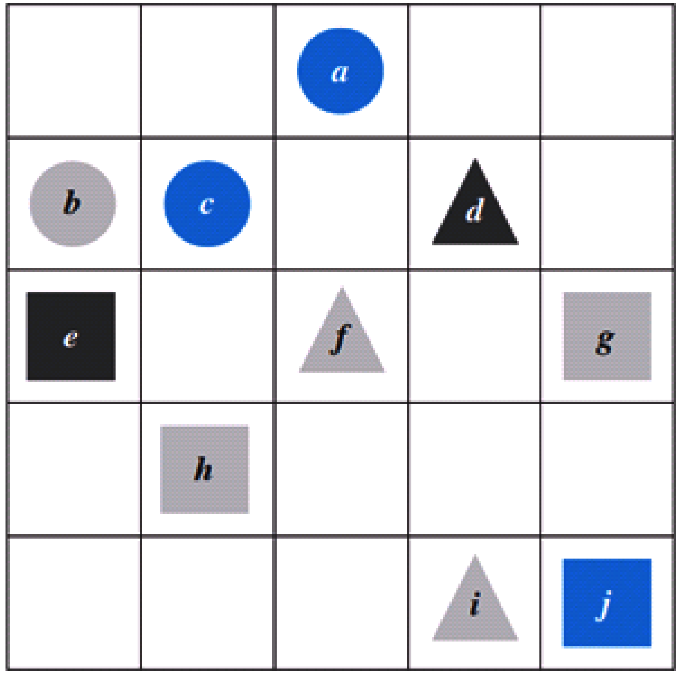
TRUE

- There exists a triangle y such that for every square x, x and y have different colors

FALSE

Statements with Multiple Quantifiers in Tarski's World

How to evaluate them?



$\exists\forall$

- There is a triangle x such that for all circles y, x is to the right of y

TRUE

Choose x =	**Then, given y =**	**check that x is to the right of y.**
d or i	a	yes ✓
	b	yes ✓
	c	yes ✓

Negations of Multiply-Quantified Statements

- Apply negation to quantified statements from left to right:

$\sim(\forall x \text{ in } D, \exists y \text{ in } E \text{ such that } P(x, y))$

$\equiv \exists x \text{ in } D \text{ such that } \sim(\exists y \text{ in } E \text{ such that } P(x, y))$

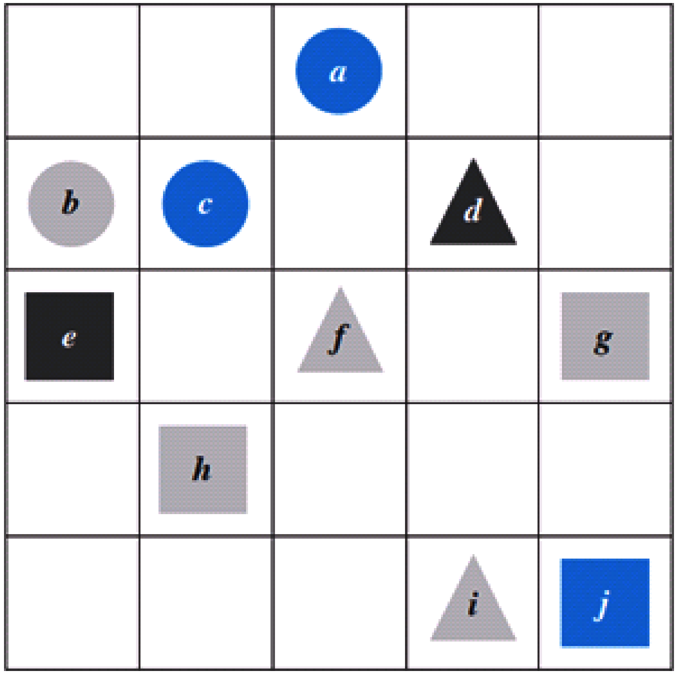
$\equiv \exists x \text{ in } D \text{ such that } \forall y \text{ in } E, \sim P(x, y).$

$\sim(\exists x \text{ in } D \text{ such that } \forall y \text{ in } E, P(x, y))$

$\equiv \forall x \text{ in } D, \sim(\forall y \text{ in } E, P(x, y))$

$\equiv \forall x \text{ in } D, \exists y \text{ in } E \text{ such that } \sim P(x, y).$

Negating Statements in Tarski's World



- For all squares x, there is a circle y such that x and y have the same color

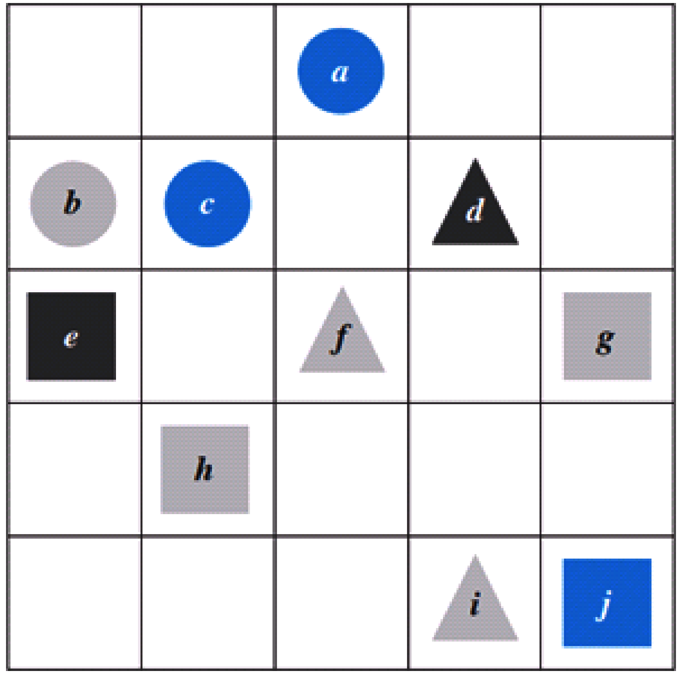
Negation:

∃ a square x such that ∼(∃ a circle y such that x and y have the same color)

≡ ∃ a square x such that ∀ circles y, x and y do not have the same color

TRUE: Square e is black and no circle is black.

Negating Statements in Tarski's World



- There is a triangle x such that for all squares y, x is to the right of y

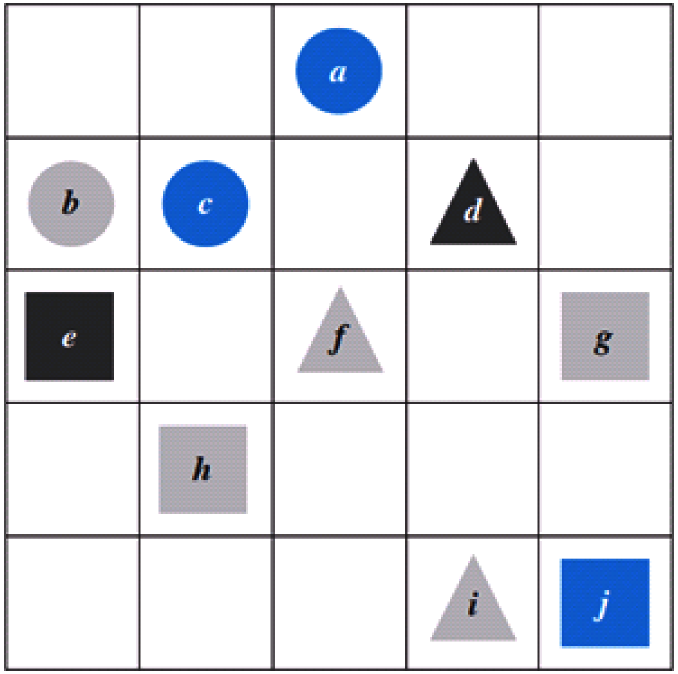
Negation:

∀ triangles x, ~ (∀ squares y, x is to the right of y)

≡ ∀ triangles x, ∃ a square y such that x is not to the right of y

TRUE

Formalizing Statements in Tarski's World



- For all circles x, x is above f

$\forall x(\mathrm{Circle}(x) \rightarrow \mathrm{Above}(x, f))$

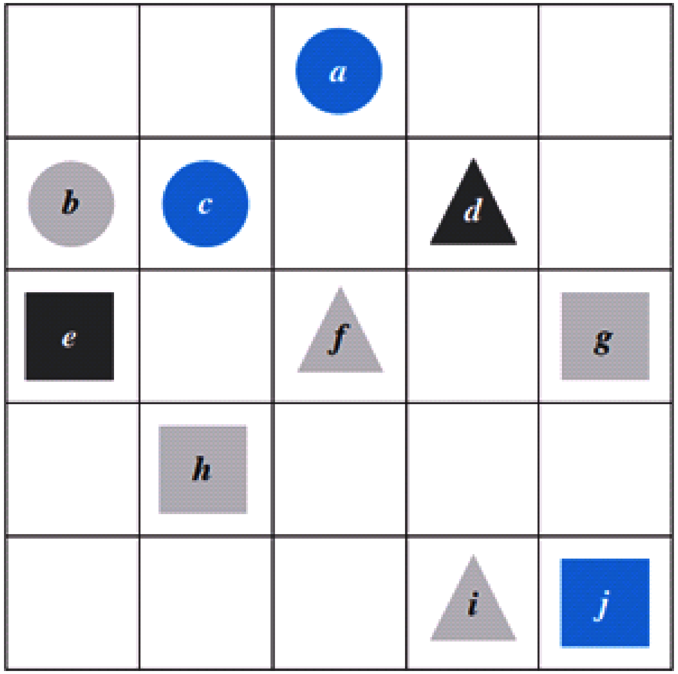
- Negation:

$\sim(\forall x(\mathrm{Circle}(x) \rightarrow \mathrm{Above}(x, f)))$

$\equiv \exists x \sim (\mathrm{Circle}(x) \rightarrow \mathrm{Above}(x, f))$

$\equiv \exists x(\mathrm{Circle}(x) \wedge \sim\mathrm{Above}(x, f))$

Formalizing Statements in Tarski's World



- There is a square x such that x is black

$\exists x(\mathrm{Square}(x) \wedge \mathrm{Black}(x))$

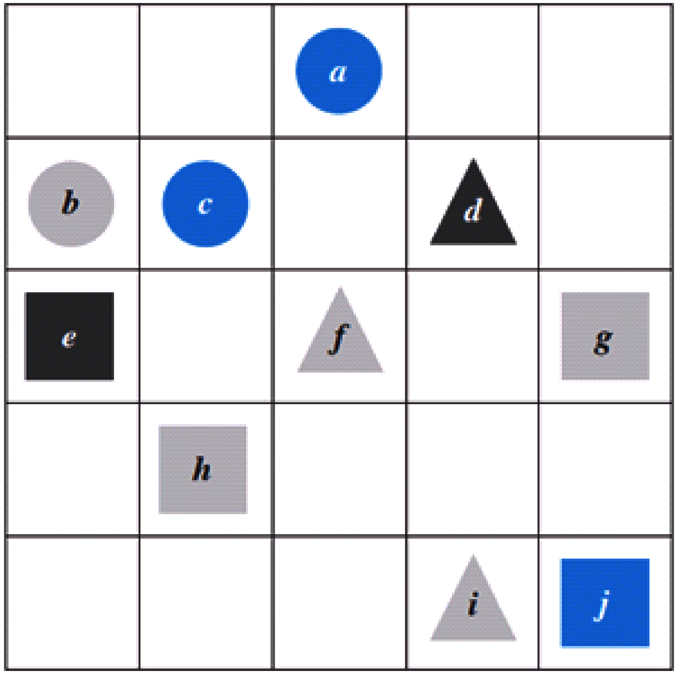
- Negation:

$\sim(\exists x(\mathrm{Square}(x) \wedge \mathrm{Black}(x)))$

$\equiv \forall x \sim (\mathrm{Square}(x) \wedge \mathrm{Black}(x))$

$\equiv \forall x(\sim\mathrm{Square}(x) \vee \sim\mathrm{Black}(x))$

Formalizing Statements in Tarski's World



- For all circles x, there is a square y such that x and y have the same color

$\forall x(\text{Circle}(x) \rightarrow \exists y(\text{Square}(y) \wedge \text{SameColor}(x, y)))$

- Negation:

$\sim(\forall x(\text{Circle}(x) \rightarrow \exists y(\text{Square}(y) \wedge \text{SameColor}(x, y))))$

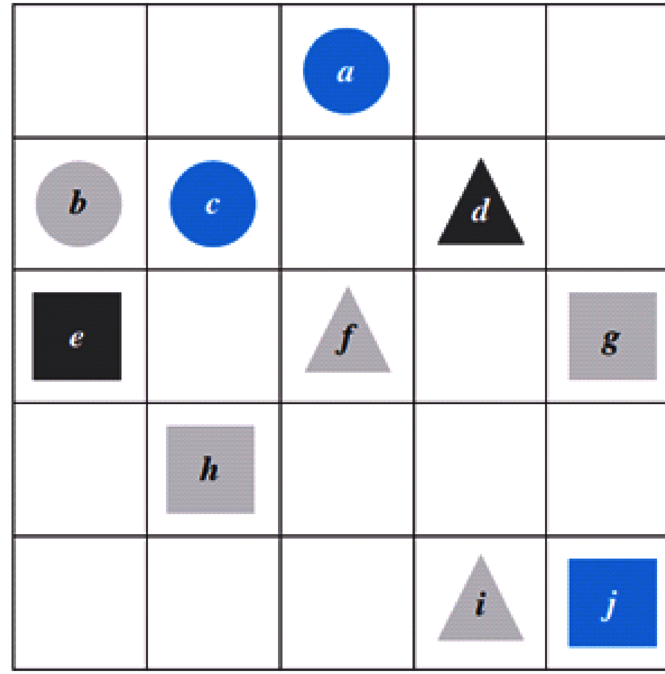
$\equiv \exists x \sim (\text{Circle}(x) \rightarrow \exists y(\text{Square}(y) \wedge \text{SameColor}(x, y)))$

$\equiv \exists x(\text{Circle}(x) \wedge \sim(\exists y(\text{Square}(y) \wedge \text{SameColor}(x, y))))$

$\equiv \exists x(\text{Circle}(x) \wedge \forall y(\sim(\text{Square}(y) \wedge \text{SameColor}(x, y))))$

$\equiv \exists x(\text{Circle}(x) \wedge \forall y(\sim\text{Square}(y) \vee \sim\text{SameColor}(x, y)))$

Formalizing Statements in Tarski's World



- There is a square x such that for all triangles y, x is to right of y

$\exists x(\text{Square}(x) \wedge \forall y(\text{Triangle}(y) \rightarrow \text{RightOf}(x, y)))$

- Negation:

$\sim(\exists x(\text{Square}(x) \wedge \forall y(\text{Triangle}(y) \rightarrow \text{RightOf}(x, y))))$

$\equiv \forall x \sim (\text{Square}(x) \wedge \forall y(\text{Triangle}(x) \rightarrow \text{RightOf}(x, y)))$

$\equiv \forall x(\sim\text{Square}(x) \vee \sim(\forall y(\text{Triangle}(y) \rightarrow \text{RightOf}(x, y))))$

$\equiv \forall x(\sim\text{Square}(x) \vee \exists y(\sim(\text{Triangle}(y) \rightarrow \text{RightOf}(x, y))))$

$\equiv \forall x(\sim\text{Square}(x) \vee \exists y(\text{Triangle}(y) \wedge \sim\text{RightOf}(x, y)))$

Validity of Arguments with Quantified Statements

- An argument form is ***valid***, if and only if, for any particular predicates substituted for the predicate symbols in the premises **if the resulting premise statements are all true, then the conclusion is also true**
- Logical arguments transfer from the propositional logic to the predicative logic: modus ponens, modus tollens, generalization, specialization

Universal Transitivity

Formal Version	*Informal Version*
$\forall x\, P(x) \rightarrow Q(x)$.	Any x that makes P(x) true makes Q(x) true.
$\forall x Q(x) \rightarrow R(x)$.	Any x that makes Q(x) true makes R(x) true.
$\therefore \forall x\, P(x) \rightarrow R(x)$.	∴ Any x that makes P(x) true makes R(x) true.

- Example from Tarski's World:

∀x, if x is a triangle, then x is blue.

∀x, if x is blue, then x is to the right of all the squares.

∴ ∀x, if x is a triangle, then x is to the right of all the squares

Logic and Programming

- Logic forms a **formal foundation** for describing relationships between entities
- In many cases, we can infer interesting consequences from these relationships
- When the inference procedure is simple enough, the descriptions of the relationships can be seen as programs
- The same set of relationships can be described in many ways: each resulting in a different "program"
- ***Logic Programming***: a framework for describing relationships such that inferences can be done efficiently

Programming Languages

- Languages:
  - Imperative = Turing machines
  - Functional Programming = lambda calculus
  - **Logical Programming** = first-order predicate calculus
- ***Prolog*** (**Pro**gramming in **log**ic) and its variants make up the most commonly used Logical programming languages.
  - One variant is XSB → developed at Stony Brook 
  - Other Prolog systems: SWI Prolog, Sicstus, Yap, Ciao, GNU Prolog, etc.

Association for Logic Programming

- http://www.cs.nmsu.edu/ALP/
  - the current state of logic programming technology
- Many other groups (start from news://comp.lang.prolog)
- XSB: http://xsb.sourceforge.net
  - system with SLG-resolution, HiLog syntax, and unification factoring
- SWI Prolog: http://www.swi-prolog.org
  - Complete, ISO and Edinburgh standard, common optimizations, GC including atoms. Portable graphics, threads, constraints, comprehensive libraries for (semantic) web programming, Unicode, source-level debugger
  - Yap Prolog: http://www.ncc.up.pt/~vsc/Yap/

Extensions of Prolog

- Flexibility of reasoning is one of the key property of intelligence.
  - ***Commonsense*** inference is ***defeasible*** in its nature: we are all capable of drawing conclusions, acting on them to derive more conclusions, and then retracting them if necessary in the face of new evidence or resulting inconsistency.
    - If computer programs are to act intelligently, they will need to be similarly flexible.

Flexible Reasoning Examples

- **Reiter, 1987:** Consider a statement *Birds fly. Tweety*, we are told, *is a bird*. From this, and the fact that birds fly, we conclude that: *Tweety can fly.*
- **This is *defeasible*:** Tweety may be an ostrich, a penguin, a bird with a broken wing, or a bird whose feet have been set in concrete.
- ***Non-monotonic* Inference:** on learning a new fact (that *Tweety has a broken wing*), we are forced to retract our conclusion (that he could fly).

Non-monotonic Logics

- ***Non-monotonic Logic*** is a logic in which the introduction of a new information (axioms) can invalidate old theorems.

Default reasoning

- ***Default reasoning*** (logics) means drawing of **plausible inferences** from less-then-conclusive evidence in the absence of information to the contrary.
  - Non-monotonic reasoning is an example of the default reasoning.

Auto-epistemic reasoning

- **Moore, 1983:** *Consider my reason for believing that I do not have an older brother. It is surely not that one of my parents once casually remarked,You know, you don't have any older brothers, nor have I pieced it together by carefully sifting other evidence.*
- *I simply believe that if I did have an older brother I would know about it; therefore, since I don't <u>know</u> of any older brothers of mine, I must not have any.*
- ***Closed-world vs. open-world assumption***

Auto-epistemic reasoning

- "The brother" reasoning is not a form of default reasoning nor non-monotonic. It is reasoning about one's **own knowledge or belief**.
  - Hence it is called an ***auto-epistemic reasoning***.
- ***Auto-epistemic reasoning*** models the reasoning of an ideally rational agent reflecting upon his beliefs or knowledge.
  - ***Auto-epistemic Logics*** are logics which describe the reasoning of an ideally rational agent reflecting upon his beliefs.

Missionaries and Cannibals

- **McCarthy, 1985** revisits the problem: Three missionaries and three cannibals come to a river. A rowboat that seats two is available. If the cannibals ever outnumber the missionaries on either bank of the river, the missionaries will be eaten. How shall they cross the river?
- Traditionally the puzzler is expected to devise a strategy of rowing the boat back and forth that gets them all across and avoids the disaster.

Missionaries and Cannibals

- Traditional Solution: A state is a triple comprising the number of missionaries, cannibals and boats on the starting bank of the river:
  - The initial state is 331, the desired state is 000.
  - A solution is given by the sequence: 331, 220, 321, 300, 311, 110, 221, 020, 031, 010, 021, 000.

Missionaries and Cannibals

- Imagine now giving someone a problem, and after he puzzles for a while, he suggests going upstream half a mile and crossing on a bridge.
  - What a bridge? you say. No bridge is mentioned in the statement of the problem.
  - He replies: Well, they don't say the isn't a bridge.

Open world assumption!

Missionaries and Cannibals

- So you modify the problem to exclude the bridges and pose it again.
- He proposes a helicopter, and after you exclude that, he proposes a winged horse or that the others hang onto the outside of the boat while two row.
- He also attacks your solution on the grounds that the boat might have a leak or lack oars.

Missionaries and Cannibals

- Finally, you must look for a mode of reasoning that will settle his hash once and for all (Closed world assumption!)
- McCarthy proposed ***circumscription***
  - He argued that it is a part of common knowledge that a boat can be used to cross the river unless there is something with it or something else prevents using it.
  - If our facts do not require that there be something that prevents crossing the river, circumscription will generate the conjecture that there isn't.
  - Lifschits has shown in 1987 that in some special cases the circumscription is equivalent to a **first order sentence** that can be added to the predicate logic program to obtain closed world

Logic Programming

- Logic Programming encompasses many types of logic:
  - Horn clauses
  - Non-monotonic
  - Constraint solving
  - Satisfiability checking
  - Knowledge Representation-Object-oriented
  - Inductive logic programming
  - Transaction Logic, Probabilistic, etc.

https://en.wikipedia.org/wiki/Logic_programming

Applications

- Deductive databases, Model checking, Declarative networking, Configuration systems, etc.
- Where? International Space Station, IBM Watson, US Border Control, Windows user access, etc.
- Conferences: International Conference on Logic Programming (ICLP), International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR), International Web Rule Symposium (RuleML) (in 2016 it was in Stony Brook), International Conference on Web Reasoning and Rule Systems (RR), etc.

Knowledge Systems Lab, Stony Brook Univ.

Paul Fodor, Michael Kifer, IV Ramakrishnan, CR Ramakrishnan, David S. Warren, Annie Liu

- Logic Programming and Deductive databases
- XSB Prolog (30+ years of research at Stony Brook)
  - http://xsb.sourceforge.net and Flora-2, LMC, ETALIS, Ergo, …
- Knowledge Representation & Processing (decision support)
- Research Interests and Projects:
  - **Logic programming: Transaction Logic, F-logic, HiLog, Defeasible Argumentation, Paraconsistency, etc.**
  - Knowledge representation
  - NLP, NLU : IBM Watson Question Analysis with Prolog, Project Halo (Vulcan Inc.) SILK
  - Rule systems benchmarking: OpenRuleBench
  - Stream processing: ETALIS/EP-SPARQL
  - Access control policies and trust management languages
  - Semantic Web
  - Virtual expert systems , …

What is Tabling?
What is Datalog?

		<u>Prolog</u>
Socrates is a man.		*man(socrates).*
All men are mortal.	<u>FOL:</u> $\forall x$, man(x)→ mortal(x).	*mortal(X) :- man(X).*
Is Socrates mortal?		*?- mortal(X).*
		Yes: X=socrates

- Prolog has goal directed top-down resolution
- The *not* (\+) operator is a *closed-world* ***negation as failure:*** if no proof can be found for the fact, then the negative goal succeeds.
  - Example: *illegal(X) :- \+ legal(X).*
  - Adding a fact that something is legal destroys an argument that it is illegal.
- Prolog's "Yes" means "I can prove it", while Prolog's "No" means "I can't prove it"

Logic Programming Extensions at Stony Brook

- Prolog pitfalls:
  - redundant computations
  - non-termination of otherwise correct programs

```
path (A , B): - path (A , C), edge (C , B).
path (A , B): - edge (A , B).
```

  - not OO, not defeasible, closed world assumption, …
- Goal: Realize the vision of logic-based knowledge representation with frames, defeasibility, meta, and side-effects, event streams, …
  - Tabling (efficiency, termination, Datalog and well-founded semantics),
  - F-logic (frames, path expressions and reification),
  - Logic programming with defaults and argumentation theories,
  - HiLog,
  - Transaction Logic (and tabling for WFS),
  - Event Condition Action rules and Complex Event Processing (ETALIS) (complex events, aggregates, consumption policies, time and count windows)

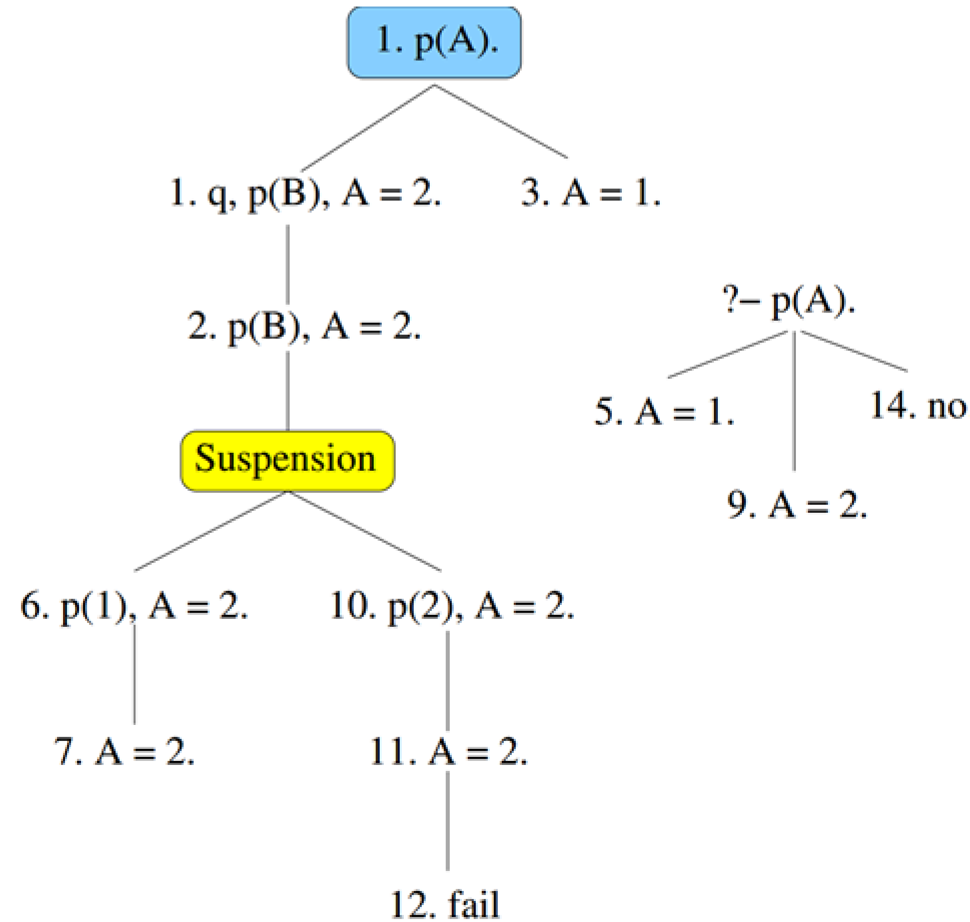
Logic Programming Extensions at Stony Brook

Suspend computation when same goal is called again and Consume answers of producers. XSB is sound and complete for LP well-founded semantics.

Example

```
:- table p/1.
q.
p(A) :-
    q,
    p(B),
    A = 2.
p(A) :-
    A = 1.
```

Subgoal	Answers
1. p(A)	4. A = 1 8. A = 2 13. Complete



Logic Programming Extensions at Stony Brook: F-Logic (Flora2)

Object description:

Object Id | *Attribute*

John[*name* –> 'John Doe', *phones* –> {6313214567, 6313214566},
children –> {Bob, Mary}]

Mary[*name* –> 'Mary Doe', *phones* –> {2121234567, 2121237645},
children –> {Anne, Alice}]

Structure can be nested:

Attribute

Sally[*spouse* –> John[*address* –> '123 Main St.']]

Methods:?P[*ageAsOf*(?Year) –> ?Age] :–
?P:Person, ?P[*born* –> ?B], ?Age is ?Year–?B.

Type signatures: Person[| *born* => \integer,
ageAsOf(\integer) => \integer |].

Logic Programming Extensions at Stony Brook: Transaction Logic

$stack(0,?X).$

$stack(?N,?X) :- ?N>0 \otimes move(?Y,?X) \otimes stack(?N-1,?Y).$

$move(?X,?Y) :- pickup(?X) \otimes putdown(?X,?Y).$

$pickup(?X) :- clear(?X) \otimes on(?X,?Y) \otimes t\_delete\{on(?X,?Y)\} \otimes t\_insert\{clear(?Y)\}.$

$putdown(?X,?Y) :- wider(?Y,?X) \otimes clear(?Y) \otimes t\_insert\{on(?X,?Y)\} \otimes t\_delete\{clear(?Y)\}.$

- Can express not only execution, but all kinds of sophisticated constraints:

  ?- $stack(10, \text{block43})$

  $\wedge \quad \forall ?X,?Y\ (move(?X,?Y) \otimes color(?X,red)) \Rightarrow (\exists\ ?Z\ color(?Z,blue) \otimes move(?Z,?X))$

  Whenever a red block is stacked, the next block to be stacked must be blue

- Planning with Heuristics: Specifying STRIPS in Transaction Logic

  $achieve\_unstack(?X,?Y) :-$

  $(achieve\_clear(?X) * achieve\_on(?X,?Y) * achieve\_handempty)$

  $\otimes\ unstack(?X,?Y).$

- Tabling to stop infinite computation paths and defeasibility (ICLP2009)

Logic Programming Extensions at Stony Brook: Defeasibility

- Common sense reasoning: rules can be true by default but may be defeated (policies, regulations, law, inductive/scientific learning, natural language understanding): Logic Programming with Defaults and Argumentation theories LPDA (ICLP2009) and Transaction Logic LPDA (ICLP2011)

```
buy :- pay ⊗ delivery.
@b1delivery :- gold_member ⊗ express_mail.
@b2delivery :- ground_mail.
@b3pay :- pay_credit_card.
@b4pay :- pay_cheque.
!opposes(b1, b2).!overrides(b1, b2).!opposes(b4, b3).!overrides(b4, b3).
express_mail :- insert(delivered_express_mail).
ground_mail :- insert(delivered_ground_mail).
pay_credit_card :- credit_card_credentials ⊗ insert(credit_card_payment).
pay_cheque :- bank_account ⊗ insert(bank_payment).
credit_card_credentials.bank_account.gold_member.
```

Argumentation theory:

```
$defeated(R)       :- $refutes(S, R) ∧ not $compromised(S).
$defeated(R)       :- $rebuts(S, R) ∧ not $compromised(S).
$defeated(R)       :- $disqualified(R).
$refutes(R, S)     :- $conflict(R, S) ∧ !overrides(R, S).
$rebuts(R, S)      :- $candidate(R) ∧ $candidate(S)∧
                      !opposes(R, S) ∧ not $compromised(R)∧
                      not $refutes(_, R) ∧ not $refutes(_, S).
$compromised(R)    :- $refuted(R) ∧ $defeated(R).
$disqualified(X)   :- $defeats_tc(X, X).
$defeats_tc(X, Y)  :- $defeats(X, Y).
$defeats_tc(X, Y)  :- $defeats_tc(X, Z) ∧ $defeats(Z, Y).
!opposes(handle(_, H), handle(_, neg H)).
```

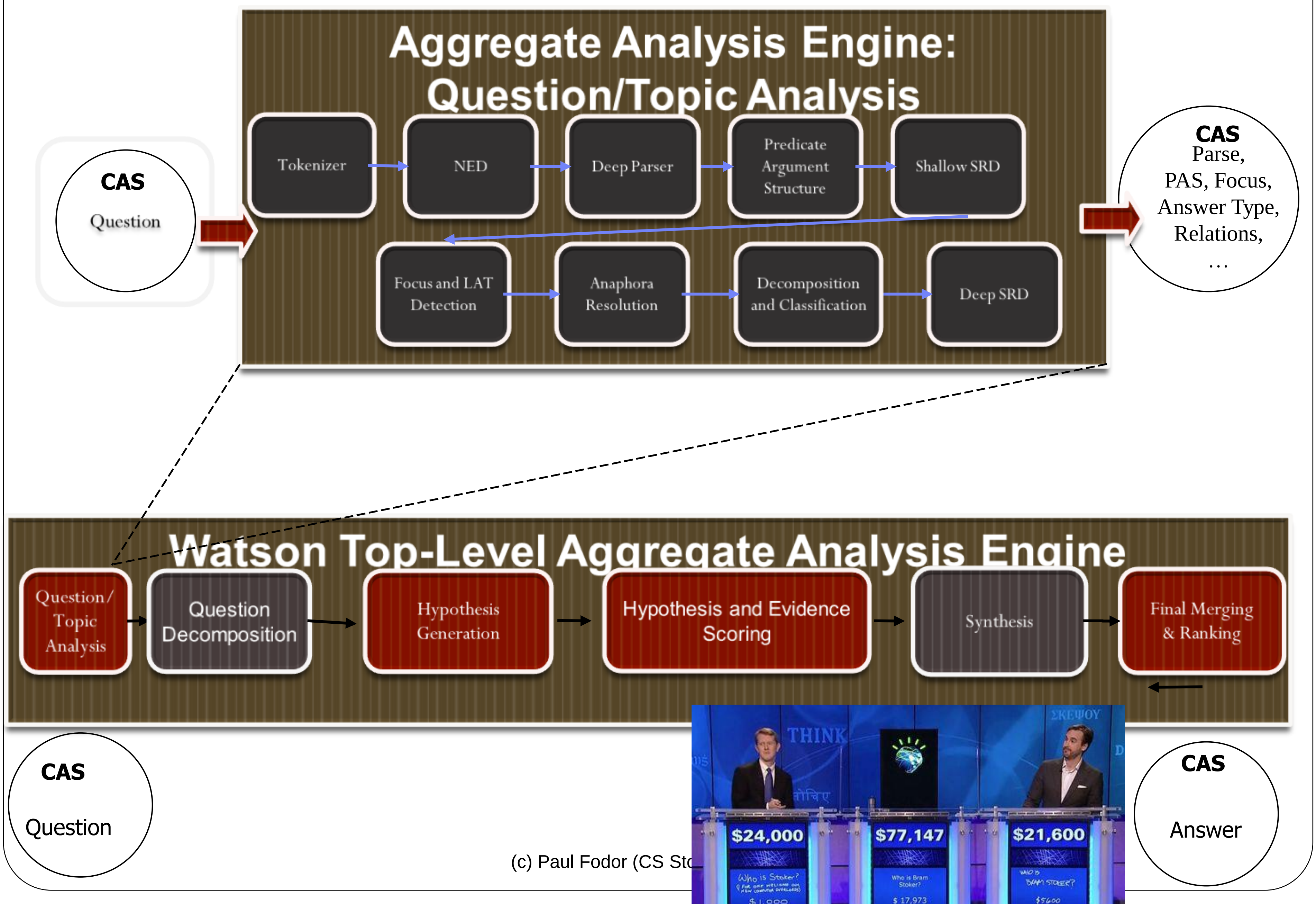
Natural Language Processing with Prolog in the IBM Watson System

- Pattern Matching: question to candidate passages
  - Coding Pattern Matching Rules Directly in a Procedural Language Like Java is Not Convenient
  - Prolog: well-established standard; straightforward syntax; very expressive; development, debugging, and profiling tools exist; efficient, well-understood implementations, proven to be effective for pattern-matching tasks; natural fit for integration with UIMA (IBM R&D Journal 2012)
- We implemented Prolog rule sets for:
  - Focus Detection
  - Lexical Answer Type Detection
  - Shallow and Deep Relation Extraction
  - Question Classification
- Execution is Efficient to Compete At Jeopardy!
  - A Question is analyzed in a fraction of a second
- Open NLP tooling at Stony Brook University

  http://ewl.cewit.stonybrook.edu/sbnlp
  - + **Education:** Stony Brook University courses:

  Computers playing Jeopardy! (2011 - 2016)

Watson Question Analysis



Focus Detection Rules

- The focus is the “node“ that refers to the unspecified answer.
- Pattern: WHAT IS X …?

  “What is the democratic party symbol?”

  “What is the longest river in the world?”

```
focus(QuestionRoot, [Pred]):-
    getDescendantNodes(QuestionRoot,Verb),
    lemmaForm(Verb,"be"),
    subj(Verb,Subj),
    lemmaForm(Subj,SubjString),
    whatWord(SubjString), % "what","which“ ("this","these“)
    pred(Verb,Pred),!.
```

- Pattern: “How much/many”:

  “How many hexagons are on a soccer ball?”

  “How much does the capitol dome weigh?”

  “How much folic acid should an expectant mother get daily?”

```
focus(QuestionRoot, [Determiner]):-
     getDescendantNodes(QuestionRoot,Determiner),
     lemmaForm(Determiner,DeterminerString),
     howMuchMany(DeterminerString),!. % "how much/many", "this much"
```

Answer-type Computation Rules

- Time rule (e.g. when): Pattern: When VERB OBJ; OBJ VERB then

Example: When was the US capitol built? answerType => [“com.ibm.hutt.Year“]

```
answerType(_QuestionRoot,FocusList,timeAnswerType,ATList):-
    member(Mod,FocusList),
    lemmaForm(Mod,ModString),
    wh_time(ModString), % "when", "then“
    whadv(Verb,Mod),
    lemmaForm(Verb,VerbString),
    timeTableLookup(VerbString,ATList),!.
```

- “How … VERB” rule: Pattern: How … VERB?

Example: “How did Virginia Woolf die?” answerType => ["com.ibm.hutt.Disease", "com.ibm.hutt.MannerOfKilling", "com.ibm.hutt.TypeOfInjury"]

```
answerType(_QuestionRoot,FocusList,howVerb1,ATList):-
     member(Mod,FocusList),
     lemmaForm(Mod,"how"),
     whadv(Verb,Mod),
     lemmaForm(Verb,VerbString),
     howVerbTableLookup(VerbString,ATList), !.
```

Answer-type Computation Rules

Focus lexicalization (lexical chains using Prolog WordNet followed by a mapping to our taxonomy)

Question	**QParse 2 AnswerType**
What American revolutionary general turned over West Point to the British?	[com.ibm.hutt.MilitaryLeader]

Table lookup for the verb:

Question	**QParse 2 AnswerType**
How did Jimi Hendrix die?	[com.ibm.hutt.Disease com.ibm.hutt.MannerOfKilling com.ibm.hutt.TypeOfInjury]

Table lookup for the focus:

Question	**QParse 2 AnswerType**
How far is it from the pitcher's mound to home plate?	[com.ibm.hutt.Length]
When was Lyndon B Johnson president?	[com.ibm.hutt.Year]

Table lookup for the focus (noun) + the verb:

Question	**QParse 2 AnswerType**
What instrument measures radioactivity?	[com.ibm.hutt.Tool]
What instrument did Louis Armstrong play?	[com.ibm.hutt.MusicalInstrument]

Answer-type Computation Rules

- Cascading rules in order of generality
  - first rule that fires returns the most specific answer-type for the question

Look at the focus + verb:

Question	QParse 2 AnswerType
How much did Marilyn Monroe weigh?	[com.ibm.hutt.Weight]
How much did the first Barbie cost?	[com.ibm.hutt.Money]

Look at the focus + noun:

Question	QParse 2 AnswerType
How many Earth days does it take for Mars to orbit the sun?	[com.ibm.hutt.Duration]
How many people visited Disneyland in 1999?	[com.ibm.hutt.Population]

Look only at the focus:

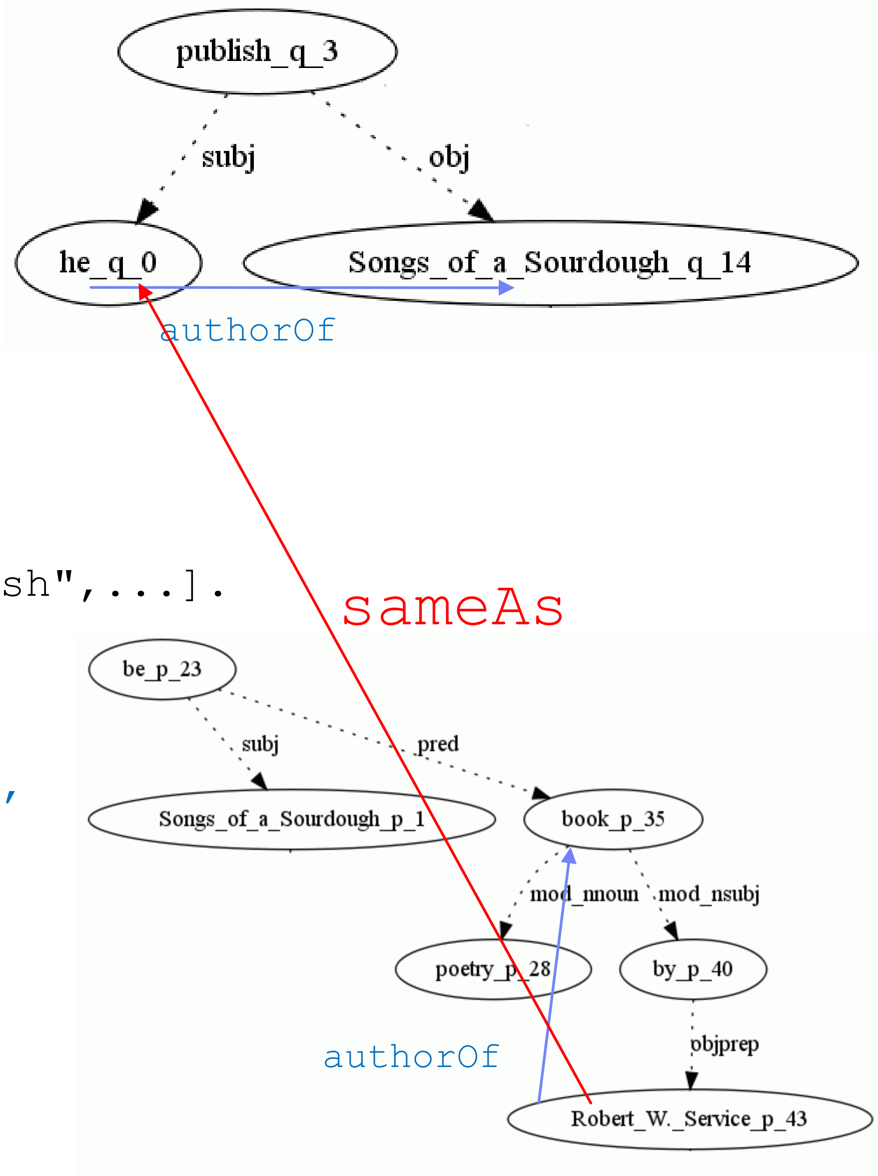
Question	QParse 2 AnswerType
How many moons does Venus have?	[com.ibm.hutt.WholeNumber]
How much calcium is in broccoli?	[com.ibm.hutt.Number]

Priority decreases down the chain

Relation Detection Rules

```
authorOf(Author,Composition) :-
  authorVerb(Verb),
  subj(Verb,Author),
  validAuthor(Author),
  obj(Verb,Composition),
  validComposition(Composition).
authorVerb(Verb) :-
  partOfSpeech(Verb,verb),
  lemma(Verb,VerbLemma),
  member(VerbLemma, ["write","publish",...].

authorOf(Author,Composition) :-
  validComposition(Composition),
  argument(Composition,Preposition),
  lemma(Preposition, "by"),
  objprep(Preposition,Author),
  validAuthor(Author).
sameAs(X,Z) :-
  authorOf(X,Y),
  authorOf(Z,Y).
```



ETALIS/ EP-SPARQL Complex Event and Stream Processing

- Data-driven continuous complex event processing:
  - Event filtering, enrichment, projection, translation, and multiplication
  - Declarative semantics
  - Combines detection of complex events and reasoning over states
  - Sliding windows (time and count-based)
  - Aggregation over events (count, avg, sum, min, max, user-defined aggregates)
  - Processing of out-of-order events
- Visual development for sequential and aggregative patterns
- Open source: http://code.google.com/p/etalis
- Uses: stock market, health applications, transit applications, NLP streaming applications (Twitter posts analysis)
  - The Ford OpenXC Challenge: map as weighted graph and update road weights from traffic events

"The Fast Flower Delivery Use Case", *accompanying the book "Event Processing In Action", by Opher Etzion and Peter Niblett, Manning Publications*

```
% Phase 1: Bid Phase
% Multiplier: multiply the event "delivery_request_enriched" for each driver
delivery_request_enriched_multiplied(DeliveryRequestId,DriverId,StoreId,ToCoordinates,DeliveryTime,
        MinRank)<-
    delivery_request_enriched(DeliveryRequestId,StoreId,ToCoordinates,
        DeliveryTime,MinRank) event_multiply driver_record(DriverId,_Ranking).
% gps_location_translated/3
gps_location_translated(DriverId,Rank,Region)<-
    gps_location(DriverId,coordinates(SNHemisphere,Latitude,EWHemisphere,Longitude)) where
        ( driver_record(DriverId,Rank),
          gps_to_region(coordinates(SNHemisphere,Latitude, EWHemisphere,Longitude),Region) ).
% bid_request/5
bid_request(DeliveryRequestId,DriverId,StoreId,ToCoordinates,DeliveryTime)<-
    delivery_request_enriched_multiplied(DeliveryRequestId,DriverId, StoreId,ToCoordinates,
        DeliveryTime, MinRank) and
    gps_location_translated(DriverId,Rank,Region)
    where MinRank <= Rank, gps_to_region(ToCoordinates,Region).
% Phase 2: Assignment Phase
startAssignment(DeliveryRequestId,StoreId,ToCoordinates, DeliveryTime) <-
    delivery_request_enriched(DeliveryRequestId,StoreId,ToCoordinates, DeliveryTime,_MinRank)
    where trigger(start_assignment_time(Time)).
assignment(DeliveryRequestId,StoreId,ToCoordinates,DeliveryTime,DriverId,ScheduledPickupTime)<-
    startAssignment(DeliveryRequestId,StoreId,ToCoordinates,DeliveryTime) and
    min(ScheduledPickupTime,
        delivery_bid(DeliveryRequestId,DriverId,CurrentCoordinates, ScheduledPickupTime) ).
```

OpenRuleBench: Analysis of the Performance of Rule Engines

Stony Brook University

- Performance tests: database tests (joins, indexing, inference), updates vs. querying, database recursion, default negation in the body, real-data tests (Mondial, DBLP, Wordnet, ontologies), AI puzzles.
- E.g., recursive stratified negation tests:

size	6000	6000	24000	24000
cyclic data	no	yes	no	yes
xsb	2.359	3.408	42.824	44.487
yap	1.875	3.148	43.510	43.452
dlv	20.274	31.346	365.136	438.008
drools	104.884	error	error	error
jess	64.000	error	1517.000	error
jena	21.007	37.692	387.268	415.376
owlim	8.666	13.314	174.968	195.825

- Systems tested: highly optimized Prolog-based systems (XSB, Yap, SWI), deductive databases (DLV, Iris, Ontobroker), rule engines for triples (Jena, BigOWLIM), production and reactive rule systems (Drools, Jess, Prova), knowledge base systems (CYC).

http://rulebench.semwebcentral.org OpenRuleBench