More valid formulas involving quantifiers:

- $\neg\forall x P(x) \Leftrightarrow \exists x \neg P(x)$
- Replacing P by $\neg P$, we get:

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

- Therefore

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

- Similarly, we have

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

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

Bound and Free Variables

$\forall i(i^2 > i)$ is equivalent to $\forall j(j^2 > j)$:

- the i and j are *bound* variables, just like the i, j in

$$\sum_{i=1}^{n} i^2 \text{ or } \sum_{j=1}^{n} j^2$$

What about $\exists i(i^2 = j)$:

- the i is bound by $\exists i$; the j is *free*. Its value is unconstrained.
- if the domain is the natural numbers, the truth of this formula depends on the value of j.

Theorems and Proofs

Just as in propositional logic, there are axioms and proof rules that provide a complete axiomatization for first-order logic, independent of the domain.

A typical axiom:

- $\forall x(P(x) \Rightarrow Q(x)) \Rightarrow (\forall x P(x) \Rightarrow \forall x Q(x))$.

Suppose we restrict the domain to the natural numbers, and allow only the standard symbols of arithmetic ($+$, $\times$, $=$, $>$, 0, 1). Typical true formulas include:

- $\forall x \exists y(x \times y = x)$
- $\forall x \exists y(x = y + y \vee x = y + y + 1)$

Let $Prime(x)$ be an abbreviation for

$$\forall y \forall z((x = y \times z) \Rightarrow ((y = 1) \vee (y = x)))$$

- $Prime(x)$ is true if x is prime

What does the following formula say:

- $\forall x(\exists y(y > 1 \wedge x = y + y) \Rightarrow$
  $\exists z_1 \exists z_2(Prime(z_1) \wedge Prime(z_2) \wedge x = z_1 + z_2))$
- This is *Goldbach's conjecture*: every even number other than 2 is the sum of two primes.
  - Is it true? We don't know.

Is there a sound and complete axiomatization for arithmetic?

- A small collection of axioms and inference rules such that every true formula of arithmetic can be proved from them
- *Gödel's Theorem:* NO!

Logic: The Big Picture

A typical logic is described in terms of

- *syntax*: what are the valid formulas
- *semantics*: under what circumstances is a formula true
- *proof theory/ axiomatization*: rules for proving a formula true

Truth and provability are quite different.

- What is provable depends on the axioms and inference rules you use
- Provability is a mechanical, turn-the-crank process
- What is true depends on the semantics

Syntax and Semantics for Propositional Logic

- syntax: start with primitive propositions and close off under $\neg$ and $\wedge$ (and $\vee$, $\Rightarrow$, $\Leftrightarrow$ if you want)
- semantics: need a truth assignment T
  - formally: a function T that maps primitive propositions to {true, false}.
  - define the truth of all formulas inductively
  - logicians write $T \models A$ if formula A is true under truth assignment T
  - typical inductive clauses:

    $T \models A \wedge B$ iff $T \models A$ and $T \models B$

    $T \models \neg A$ iff $T \not\models A$

Tautologies and Valid Arguments

When is an argument

A_1
A_2
$\vdots$
A_n
———
B

valid?

- When the truth of the premises imply the truth of the conclusion

How do you check if an argument is valid?

- Method 1: Take an arbitrary truth assignment v. Show that if $A_1, \ldots, A_n$ are true under T ($T \models A_1$, $\ldots v \models A_n$) then B is true under T.
- Method 2: Show that $A_1 \wedge \ldots \wedge A_n \Rightarrow B$ is a tautology (essentially the same thing)
  - true for every truth assignment
- Method 3: Try to prove $A_1 \wedge \ldots \wedge A_n \Rightarrow B$ using a sound axiomatization

A Sound and Complete Axiomatization for Propositional Logic

All you need are two axioms *schemes*:

Ax1. $A \Rightarrow (B \Rightarrow A)$

Ax2. $(A \Rightarrow (B \Rightarrow C) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$

and one inference rule: Modus Ponens:

- From $A \Rightarrow B$ and A infer B

Ax1 and Ax2 are axioms schemes:

- each one encodes an infinite set of axioms (obtained by plugging in arbitrary formulas for A, B, C

A *proof* is a sequence of formulas $A_1, A_2, A_3, \ldots$ such that each A_i is either

1. An instance of Ax1 and Ax2
2. Follows from previous formulas by applying MP
   - that is, there exist A_j, A_k with $j, k < i$ such that A_j has the form $A \Rightarrow B$, A_k is A and A_i is B.

This axiomatization is sound and complete.

- everything provable is a tautology
- all tautologies are provable

First-Order Logic: Semantics

How do we decide if a first-order formula is true? Need:

- a domain D (what are you quantifying over)
- an *interpretation* I that interprets the constants and predicate symbols:
  - for each constant symbol c, $I(c) \in D$
    - Which domain element is Alice?
  - for each unary predicate P, $I(P)$ is a predicate on domain D
    - formally, $I(P)(d) \in \{\text{true,false}\}$ for each $d \in D$
    - Is Alice Tall? How about Bob?
  - for each binary predicate Q, $I(Q)$ is a predicate on $D \times D$:
    - formally, $I(Q)(d_1, d_2) \in \{\text{true,false}\}$ for each $d_1, d_2 \in D$
    - Is Alice taller than Bob?
- a valuation V associating with each variable x and element $V(x) \in D$.
  - To figure out if $P(x)$ is true, you need to know what x is.

Now we can define whether a formula A is true, given a domain D, an interpretation I, and a valuation V, written

$$(I, D, V) \models A$$

The definition is by induction:

$(I, D, V) \models P(x)$ if $I(P)(V(x)) = \text{true}$
$(I, D, V) \models P(c)$ if $I(P)(I(c)) = \text{true}$
$(I, D, V) \models \forall x A$ if $(I, D, V') \models A$ for all valuations V' that agree with V except possibly on x

- $V'(y) = V(y)$ for all $y \neq x$
- $V'(x)$ can be arbitrary

$(I, D, V) \models \exists x A$ if $(I, D, V') \models A$ for some valuation V' that agrees with V except possibly on x.

Axiomatizing First-Order Logic

There's also an elegant complete axiomatization for first-order logic.

- Again, the only inference rule is Modus Ponens
- Typical axiom:

$$\forall x(P(x) \Rightarrow Q(x)) \Rightarrow (\forall x P(x) \Rightarrow \forall x Q(x))$$

- Completeness was proved by Gödel in 1930

Some Bureuacracy

- The final is on Thursday, May 13, 12-2:30 PM, in Philips 101
- If you have conflicts (more than two exams in a 24-hour time period) let me know as soon as possible.
  - We may schedule a makeup; or perhaps the other course will.
- Office hours go on as usual during study week, but check the course web site soon.
  - There may be small changes to accommodate the TAs exams
- There will be a review session

Coverage of Final

- everything covered by the first prelim
  - emphasis on more recent material
- Chapter 4: Fundamental Counting Methods
  - Basic methods: sum rule, product rule, division rule
  - Permutations and combinations
  - Combinatorial identities (know Theorems 1–4 on pp. 310–314)
  - Pascal's triangle
  - Binomial Theorem (but not multinomial theorem)
  - Balls and urns
  - Inclusion-exclusion
  - Pigeonhole principle
- Chapter 6: Probability:
  - 6.1–6.5 (but not inverse binomial distribution)
  - basic definitions: probability space, events
  - conditional probability, independence, Bayes Thm.
  - random variables

- uniform, binomial, and Poisson distributions
- expected value and variance
- Markov + Chebyshev inequalities
- understanding Law of Large Numbers, Central Limit Theorem

- Chapter 7: Logic:
  - 7.1–7.4, 7.6; *not* 7.5
  - translating from English to propositional (or first-order) logic
  - truth tables and axiomatic proofs
  - algorithm verification
  - first-order logic

Ten Powerful Ideas

- **Counting**: Count without counting (*combinatorics*)
- **Induction**: Recognize it in all its guises.
- **Exemplification**: Find a sense in which you can try out a problem or solution on small examples.
- **Abstraction**: Abstract away the inessential features of a problem.
  - One possible way: represent it as a graph
- **Modularity**: Decompose a complex problem into simpler subproblems.
- **Representation**: Understand the relationships between different possible representations of the same information or idea.
  - Graphs vs. matrices vs. relations
- **Refinement**: The best solutions come from a process of repeatedly refining and inventing alternative solutions.
- **Toolbox**: Build up your vocabulary of abstract structures.

- **Optimization**: Understand which improvements are worth it.
- **Probabilistic methods**: Flipping a coin can be surprisingly helpful!

Connections: Random Graphs

Suppose we have a random graph with n vertices. How likely is it to be connected?

- What is a *random* graph?
  - If it has n vertices, there are $C(n, 2)$ possible edges, and $2^{C(n,2)}$ possible graphs. What fraction of them is connected?
  - One way of thinking about this. Build a graph using a random process, that puts each edge in with probability 1/2.

- Given three vertices a, b, and c, what's the probability that there is an edge between a and b and between b and c? 1/4
- What is the probability that there is no path of length 2 between a and c? $(3/4)^{n-2}$
- What is the probability that there is a path of length 2 between a and c? $1 - (3/4)^{n-2}$
- What is the probability that there is a path of length 2 between a and every other vertex? $> (1-(3/4)^{n-2})^{n-1}$

Now use the binomial theorem to compute $(1-(3/4)^{n-2})^{n-1}$

$$\begin{aligned} & (1-(3/4)^{n-2})^{n-1} \\ = \; & 1-(n-1)(3/4)^{n-2}+C(n-1,2)(3/4)^{2(n-2)}+\cdots \end{aligned}$$

For sufficiently large n, this will be (just about) 1.

Bottom line: If n is large, then it is almost certain that a random graph will be connected.

Theorem: [Fagin, 1976] If P is *any* property expressible in first-order logic, it is either true in almost all graphs, or false in almost all graphs.

This is called a *0-1 law*.

Connection: First-order Logic

Suppose you wanted to query a database. How do you do it?

Modern database query language date back to SQL (structured query language), and are all based on first-order logic.

- The idea goes back to Ted Codd, who invented the notion of relational databases.

Suppose you're a travel agent and want to query the airline database about whether there are flights from Ithaca to Santa Fe.

- How are cities and flights between them represented?
- How do we form this query?

You're actually asking whether there is a path from Ithaca to Santa Fe in the graph.

- This fact cannot be expressed in first-order logic!