First–Order Logic

This is the most powerful, most expressive logic that we will examine.

Our version of first-order logic will use the following symbols:

- variables
- connectives $(\vee, \wedge, \rightarrow, \leftrightarrow, \neg)$
- function symbols
- relation symbols
- constant symbols
- equality $(\approx)$
- quantifiers $(\forall, \exists)$

Formulas for a first-order language $\mathcal{L}$ are defined inductively as follows:

- There are two kinds of **atomic** formulas:

  $\boxed{(s \approx t)}$, where s and t are terms, and

  $\boxed{(rt_1 \cdots t_n)}$, where r is an n–ary relation symbol and $t_1, \cdots, t_n$ are terms.

- If F is a formula, then so is $\boxed{(\neg \mathsf{F})}$.

- If F and G are formulas, then so are

$\boxed{(\mathsf{F} \vee \mathsf{G})}$, $\boxed{(\mathsf{F} \wedge \mathsf{G})}$, $\boxed{(\mathsf{F} \rightarrow \mathsf{G})}$, $\boxed{(\mathsf{F} \leftrightarrow \mathsf{G})}$.

- If F is a formula and x is a variable, then

$\boxed{(\forall x\, \mathsf{F})}$ and $\boxed{(\exists x\, \mathsf{F})}$ are formulas.

Notational Conventions

- Drop outer parentheses
- Adopt the previous precedence conventions for the propositional connectives.
- Quantifiers bind more strongly than any of the connectives.

Thus $\forall y\,(rxy) \vee \exists y\,(rxy)$

means $(\forall y\,(rxy)) \vee (\exists y\,(rxy))$.

The **subformulas** of a formula F :

- The only subformula of an atomic formula $\boxed{F}$ is F itself.

- The subformulas of $\boxed{\neg F}$ are $\neg F$ itself and all the subformulas of F.

- The subformulas of $\boxed{F \Box G}$ are $F \Box G$ itself and all the subformulas of F and all the subformulas of G. ($\Box$ is any of $\vee$, $\wedge$, $\rightarrow$, r $\leftrightarrow$).

- The subformulas of $\boxed{\forall x\, F}$ are $\forall x\, F$ itself and all the subformulas of F.

- The subformulas of $\boxed{\exists x\, F}$ are $\exists x\, F$ itself and all the subformulas of F.

An **occurrence** of a variable x in a formula F is:

• **bound** if the occurrence is in a subformula of the form $\forall x\,\mathsf{G}$ or of the form $\exists x\,\mathsf{G}$

(such a subformula is called the **scope** of the quantifier that begins the subformula).

• Otherwise the occurrence of the variable is said to be **free**.

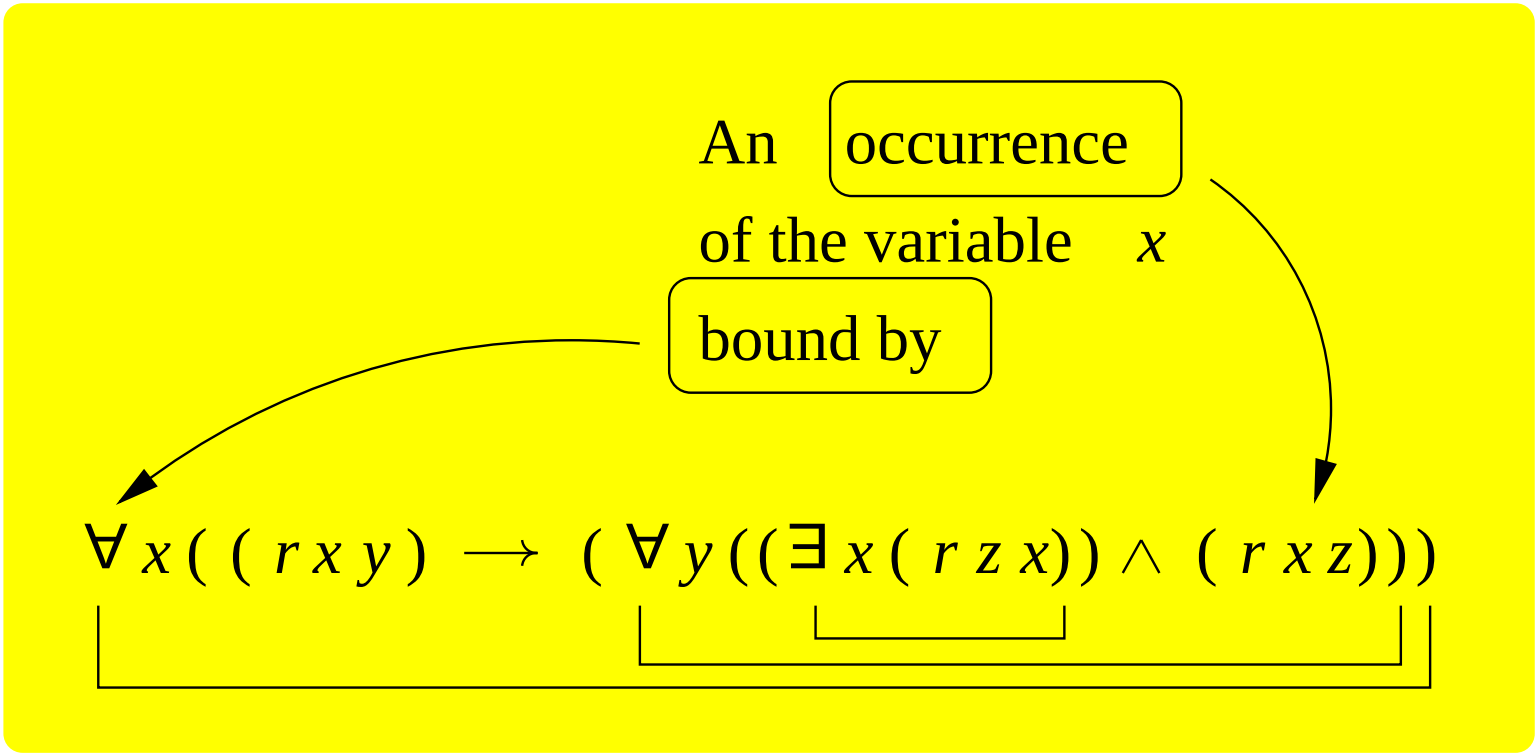
• A formula with no free occurrences of variables is called a **sentence**.

Given a bound occurrence of x in F, we say that x is **bound by** an occurrence of a quantifier Q if

(i) the occurrence of Q quantifies the variable x, and

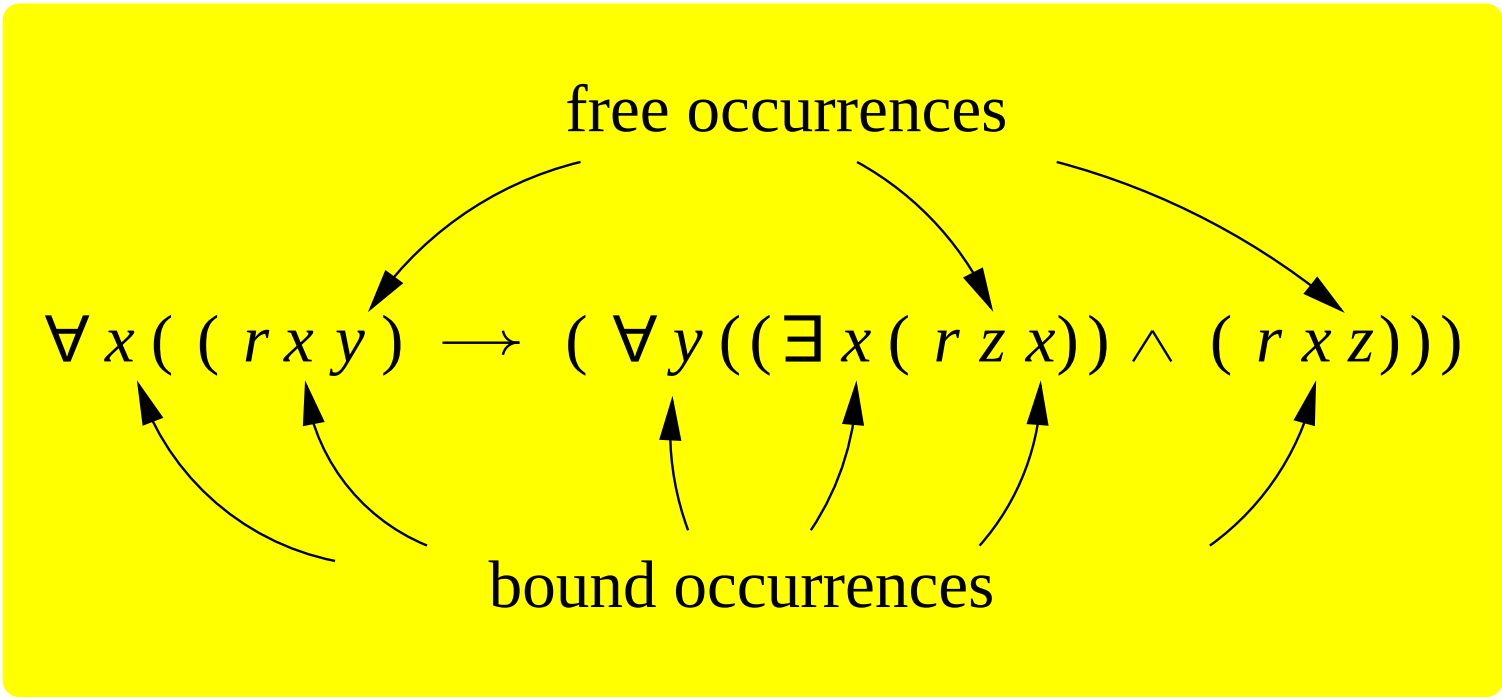
(ii) subject to this constraint the scope of this occurrence of Q is the smallest in which the given occurrence of x occurs.

It is easier to explain scope, and quantifiers that bind variables, with a diagram:



Scopes of quantifiers are underlined

The following figure indicates all the bound and free variables in the previous formula:



Examples

b. $\boxed{2+2<3}$ is also an atomic sentence, which says "four is less than three."

False in $\mathbf{N}$.

c. $\boxed{\forall x \exists y\,(x<y)}$ says that "for every number there is a larger number."

True in $\mathbf{N}$.

d. $\boxed{\exists y \forall x\,(x<y)}$ says that "there is a number that is larger than every other number."

False in $\mathbf{N}$.

f. $\boxed{\forall x \Big((0 < x) \to \exists y \, (y \cdot y \approx x) \Big)}$ says that "every positive number is a square."

False in $\mathbf{N}$.

h. $\boxed{\forall x \forall y \Big((x < y) \to \exists z \Big((x < z) \wedge (z < y) \Big) \Big)}$ says that "if one number is less than another, then there is a number properly between the two."

False in $\mathbf{N}$.

We will use the shorthand notation

$$\bigwedge_{1 \leq i \leq n} \mathsf{F}_i$$

to mean the same as the notation

$$\mathsf{F}_1 \wedge \cdots \wedge \mathsf{F}_n.$$

Likewise, we will use the notations

$$\bigvee_{1 \leq i \leq n} \mathsf{F}_i$$

and

$$\mathsf{F}_1 \vee \cdots \vee \mathsf{F}_n.$$

English to First-order

Given a first–order formula $\mathsf{F}(x)$ we can find first–order sentences to say

a. There is at least one number such that $\mathsf{F}(x)$ is true in $\mathbf{N}$.

$$\boxed{\exists x\, \mathsf{F}(x)}$$

b. There are at least two numbers such that $\mathsf{F}(x)$ is true in $\mathbf{N}$.

$$\boxed{\exists x \exists y \left(\neg\, (x \approx y) \wedge \mathsf{F}(x) \wedge \mathsf{F}(y) \right)}$$

c. There are at least n numbers (n fixed) such that $\mathsf{F}(x)$ is true in $\mathbf{N}$.

$$\boxed{\exists x_1 \cdots \exists x_n \left(\left(\bigwedge_{1 \leq i < j \leq n} \neg (x_i \approx x_j) \right) \wedge \left(\bigwedge_{1 \leq i \leq n} \mathsf{F}(x_i) \right) \right)}$$

d. There are infinitely many numbers that make $\mathsf{F}(x)$ true in $\mathbf{N}$.

$$\boxed{\forall x \exists y \left((x < y) \wedge \mathsf{F}(y) \right)}$$

e. There is at most one number such that $\mathsf{F}(x)$ is true in $\mathbf{N}$.

$$\boxed{\forall x \forall y \left((\mathsf{F}(x) \wedge \mathsf{F}(y)) \rightarrow (x \approx y) \right)}$$

Definable Relations

To better understand what we can express with first–order sentences we need to introduce definable relations.

Given a first–order formula $\mathsf{F}(x_1, \ldots, x_k)$ we say

F is "true" at a k–tuple $(a_1, \ldots, a_k)$ of natural numbers

if the expression $\mathsf{F}(a_1, \ldots, a_k)$ is a true statement about the natural numbers.

Example

Let F(x, y) be the formula $x < y$.

Then F is true at (a, b) iff a is less than b.

Example

Let F(x, y) be $\exists z\,(x \cdot z \approx y)$.

Then F is true at (a, b) iff a divides b, written $a|b$.

[**Note:** Don't confuse $a|b$ with $\frac{a}{b}$.
The first is true or false.
The second has a value.
Check that $a|0$ for any a, including $a = 0$.]

Definition

For $\mathsf{F}(x_1, \ldots, x_k)$ a formula let $\mathsf{F}^{\mathbf{N}}$ be the set of k–tuples $(a_1, \ldots, a_k)$ of natural numbers for which $\mathsf{F}(a_1, \ldots, a_k)$ is true in $\mathbf{N}$. $\mathsf{F}^{\mathbf{N}}$ is **the relation on N defined by** F.

Definition

A k–ary relation $r \subseteq N^k$ is **definable** in $\mathbf{N}$ if there is a formula $\mathsf{F}(x_1, \ldots, x_k)$ such that

$$r \;=\; \mathsf{F}^{\mathbf{N}} \;.$$

Example (Definable Relations)

- $\boxed{x \text{ is an even number}}$ is definable in $\mathbf{N}$ by

$$\exists y\,(x \approx y + y)$$

- $\boxed{x \text{ divides } y}$ is definable in $\mathbf{N}$ by

$$\exists z\,(x \cdot z \approx y)$$

- $\boxed{x \text{ is prime}}$ is definable in $\mathbf{N}$ by

$$(1 < x) \wedge \forall y \Big((y|x) \rightarrow \big((y \approx 1) \vee (y \approx x) \big) \Big)$$

- $\boxed{x \equiv y \text{ modulo } n}$ is definable in $\mathbf{N}$ by

$$\exists z \Big((x \approx y + n \cdot z) \vee (y \approx x + n \cdot z) \Big)$$

We will adopt the following abbreviations:

- $x|y$ for the formula $\boxed{\exists z\,(x \cdot z \approx y)}$

- $prime(x)$ for the formula

$$\boxed{(1 < x) \wedge \forall y \Big((y|x) \rightarrow \big((y \approx 1) \vee (y \approx x) \big) \Big)}.$$

Note that in the definition of $prime(x)$ we have used the previous abbreviation.

To properly write $prime(x)$ as a first–order formula we need to replace that abbreviation; doing so gives us

$$(1 < x) \wedge \forall y \Big(\exists z\,(y \cdot z \approx x) \rightarrow \big((y \approx 1) \vee (y \approx x) \big) \Big)$$

Abbreviations are **not** a feature of first–order logic, but rather they are a tool used by people to discuss first–order logic.

Why do we use abbreviations? Without them, writing out the first–order sentences that we find interesting would fill up lines with tedious, hard–to–read symbolism.

Given: $x|y$ abbreviates $\boxed{\exists z\,(x \cdot z \approx y)}$.

Then $(u+1)|(u \cdot u+1)$ is an abbreviation for

$$\exists z\,((u+1) \cdot z \approx u \cdot u+1)$$

Now let us write out $z|2$ to obtain

$$\exists z\,(z \cdot z \approx 1+1)$$

Unfortunately, this last formula does not define the set of elements in N that divide 2. It is a first–order sentence that is simply false in $\mathbf{N}$—the square root of 2 is not a natural number.

We have stumbled onto one of the subtler points of first–order logic, namely, we must **be careful with substitution**.

The remedy for defining "z divides 2" is to use another formula, like

$$\exists w\,(x \cdot w \;\approx\; y),$$

for "x divides y."

We obtain such a formula by simply renaming the bound variable z in the formula for $x|y$.

With this formula we can correctly express "z divides 2" by $\exists w\,(z \cdot w \approx 2)$.

The danger in using abbreviations in first–order logic is that we forget the names of the bound variables in the abbreviation.

Our solution: add a $\star$ to the abbreviation to alert the reader to the necessity for renaming the bound variables that overlap with the variables in the term to be substituted into the abbreviation.

For example, $prime^{\star}(y+z)$ alerts the reader to the need to change the formula for $prime(x)$, say to

$$(1 < x) \wedge \forall v \Big((v|x) \rightarrow \big((v \approx 1) \vee (v \approx x) \big) \Big),$$

so that when we substitute $y+z$ for x in the formula, no new occurrence of y or z becomes bound.

Thus we could express $prime(y+z)$ by

$$(1 < y+z) \wedge \forall v \Big((v|^\star(y+z)) \rightarrow \big((v \approx 1) \vee (v \approx y+z)\big)\Big).$$

Examples
[Expressing statements in first-order logic.]

a. The relation "divides" is transitive.

$$\boxed{\forall x \forall y \forall z \Big(\big((x|y) \wedge (y|^\star z) \big) \rightarrow (x|^\star z) \Big)}$$

b. There are an infinite number of primes.

$$\boxed{\forall x \exists y \Big((x < y) \wedge prime^\star(y) \Big)}$$

d. There are an infinite number of pairs of primes that differ by the number 2.
(**Twin Prime Conjecture**)

$$\forall x \exists y \Big((x < y) \wedge prime^{\star}(y) \wedge prime^{\star}(y+2) \Big)$$

e. All even numbers greater than two are the sum of two primes.
(**Goldbach's Conjecture**)

$$\forall x \Big(\Big((2|x) \wedge (2 < x) \Big) \rightarrow$$

$$\exists y \exists z \Big(prime^{\star}(y) \wedge prime(z) \wedge (x \approx y + z) \Big) \Big)$$

Other Number Systems

Our first–order language $\mathcal{L} = \{+, \cdot, <, 0, 1\}$ can just as easily be used to study other number systems, in particular,

- the integers $\mathbf{Z} = (Z, +, \cdot, <, 0, 1)$,
- the rationals $\mathbf{Q} = (Q, +, \cdot, <, 0, 1)$,
- and the reals $\mathbf{R} = (R, +, \cdot, <, 0, 1)$.

However, first–order sentences that are true in one can be false in another.

Example

Consider the following first–order sentences:

a. $\forall x \exists y \, (x < y)$

b. $\forall y \exists x \, (x < y)$

c. $\forall x \forall y \Big((x < y) \rightarrow \exists z \, \Big((x < z) \wedge (z < y) \Big) \Big)$

d. $\forall x \exists y \Big((0 < x) \rightarrow (x \approx y \cdot y) \Big)$

e. $\exists x \forall y \, (x < y)$.

Then we have

	$\mathbf{N}$	$\mathbf{Z}$	$\mathbf{Q}$	$\mathbf{R}$
a.	true	true	true	true
b.	false	true	true	true
c.	false	false	true	true
d.	false	false	false	true
e.	false	false	false	false

We arrive at the simple conclusion that the notion of truth depends on the structure being considered!

First–Order Syntax for (Directed) graphs

The first–order language of (directed) graphs is $\mathcal{L} = \{r\}$, where r is a binary relation symbol.

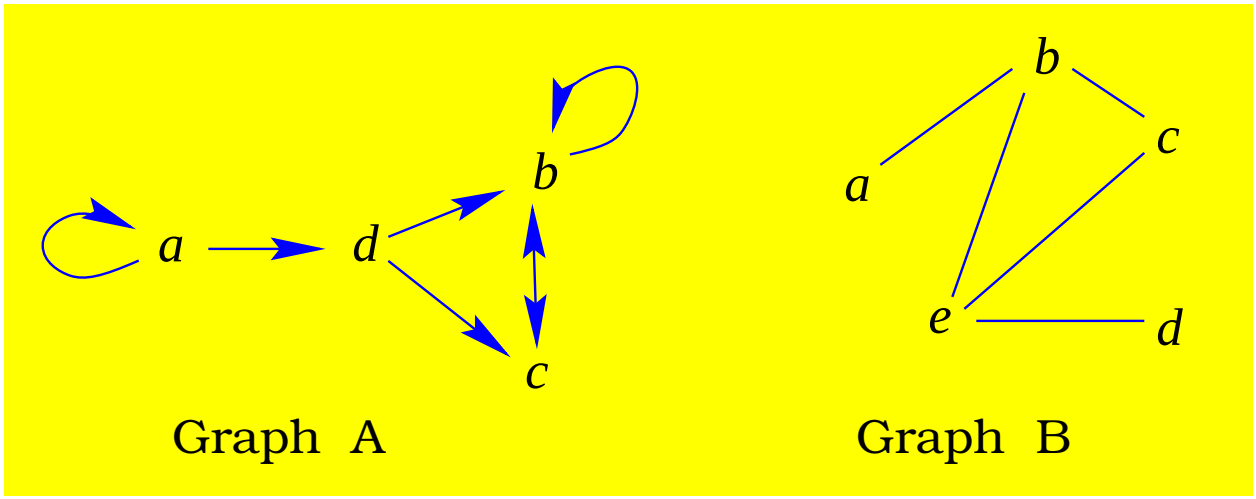
The only **terms** are the variables x.

Atomic formulas look like $(x \approx y)$ or (rxy).

Example
The subformulas of $\forall x\,((rxy) \to \exists y\,(ryx))$ are

$$\begin{array}{ll} \forall x\,((rxy) \to \exists y\,(ryx)) & rxy \\ (rxy) \to \exists y\,(ryx) & \exists y\,(ryx) \\ & ryx. \end{array}$$

First–Order to English



b. $\forall x \neg (rxx)$

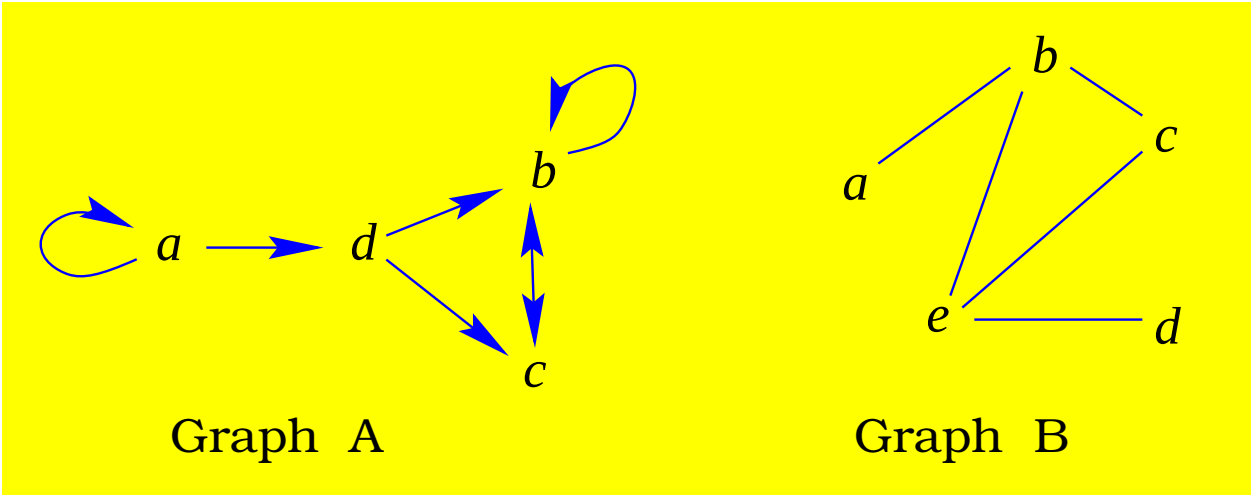
says "the (directed) graph is **irreflexive**."

False for Graph A, true for Graph B.

c. $\forall x \forall y \Big((rxy) \rightarrow (ryx) \Big)$

says "the (directed) graph is **symmetric**."

False for Graph A, true for Graph B.



e. $\forall x \forall y\,(rxy)$

says "all possible edges are present."

False for both examples.

f. $\forall x \exists y\,(rxy)$

says "for every vertex x there is an edge going from x to some vertex y."

True for both examples.

English to First-Order

a. The (directed) graph has at least two vertices.

$$\boxed{\exists x \exists y \left(\neg \left(x \approx y\right)\right)}.$$

b. Every vertex has an edge attached to it.

$$\boxed{\forall x \exists y \Big(\left(rxy\right) \vee \left(ryx\right)\Big)}.$$

c. Every vertex has at most two edges directed from it to other vertices.

$$\boxed{\forall x \forall y \forall z \forall w \Big(\Big(\left(rxy\right) \wedge \left(rxz\right) \wedge \left(rxw\right)\Big) \rightarrow \boxed{\Big(\left(y \approx z\right) \vee \left(y \approx w\right) \vee \left(w \approx z\right)\Big)\Big)}}.$$

Statements About Graphs

- The **degree** of a vertex is the number of (undirected) edges attached to it.

- A **path of length n** from vertex x to vertex y is a sequence of vertices $a_1, \ldots, a_{n+1}$ with each (a_i, a_{i+1}) being an edge, and with $x = a_1$, $y = a_{n+1}$.

- Two vertices are **adjacent** if there is an edge connecting them.

Some Definable Relations

a. The degree of x is at least one.

$$\boxed{\exists y\,(rxy)}$$

b. The degree of x is at least two.

$$\boxed{\exists y \exists z\,(\neg\,(y \approx z) \wedge (rxy) \wedge (rxz))}$$

Statements About Graphs

a. Some vertex has degree at least two.

$$\boxed{\exists x \exists y \exists z\,\Big(\neg\,(y \approx z) \wedge (rxy) \wedge (rxz)\Big)}$$

b. Every vertex has degree at least two.

$$\boxed{\forall x \exists y \exists z\,\Big(\neg\,(y \approx z) \wedge (rxy) \wedge (rxz)\Big)}$$

Semantics for First–Order Logic

Given a first–order $\mathcal{L}$–structure $\mathbf{S} = (S, I)$, the interpretation I gives meaning to the symbols of the language $\mathcal{L}$.

We associate with each term $t(x_1, \cdots, x_n)$ the n–ary term function $t^{\mathbf{S}}$.

We associate with each formula $\mathsf{F}(x_1, \cdots, x_n)$ an n–ary relation $\mathsf{F}^{\mathbf{S}} \subseteq S^n$:

[The superscript $\mathbf{S}$ is omitted in the following for ease of reading.]

- $\mathsf{F}(\vec{x})$ is atomic:

  F is $\boxed{t_1(\vec{x}\,) \approx t_2(\vec{x}\,)}$:

  $\mathsf{F}(\vec{a})$ holds iff $t_1(\vec{a}) = t_2(\vec{a})$

  F is $\boxed{r t_1(\vec{x}) \cdots t_n(\vec{x})}$

  $\mathsf{F}(\vec{a})$ holds iff $r(t_1(\vec{a}), \ldots, t_n(\vec{a}))$ holds.

- F is $\boxed{\neg\, \mathsf{G}}$

Then $\mathsf{F}(\vec{a}\,)$ holds iff $\mathsf{G}(\vec{a})$ does not hold.

- F is $\boxed{G \vee H}$

Then $F(\vec{a})$ holds iff $G(\vec{a})$ or $H(\vec{a})$ holds.

- F is $\boxed{G \wedge H}$

Then $F(\vec{a})$ holds iff $G(\vec{a})$ and $H(\vec{a})$ holds.

- F is $\boxed{G \rightarrow H}$

Then $F(\vec{a})$ holds iff $G(\vec{a})$ does not hold, or $H(\vec{a})$ holds.

- F is $\boxed{\mathsf{G} \leftrightarrow \mathsf{H}}$

Then $\mathsf{F}(\vec{a}\,)$ holds iff both or neither of $\mathsf{G}(\vec{a}\,)$ and $\mathsf{H}(\vec{a}\,)$ holds.

- $\mathsf{F}(\vec{x})$ is $\boxed{\forall y\, \mathsf{G}(y, \vec{x})}$

Then $\mathsf{F}(\vec{a}\,)$ holds iff $\mathsf{G}(b, \vec{a}\,)$ holds for **every** $b \in S$.

- $\mathsf{F}(\vec{x})$ is $\boxed{\exists y\, \mathsf{G}(y, \vec{x})}$

Then $\mathsf{F}(\vec{a}\,)$ holds iff $\mathsf{G}(b, \vec{a}\,)$ holds for **some** $b \in S$.

Example

$\mathbf{S} = \{a, b\}$

r	a	b
a	0	1
b	1	0

	f
a	b
b	a

$$\mathsf{F}(x) = \forall y \exists z\, ((rfxfy) \wedge (rfyfz)).$$

$\boxed{\mathsf{F}^{\mathbf{S}} =?}$ Let

$$\begin{aligned}
\mathsf{H}_1(x,y,z) &= rfxfy \\
\mathsf{H}_2(x,y,z) &= rfyfz \\
\mathsf{H}(x,y,z) &= \mathsf{H}_1(x,y,z) \wedge \mathsf{H}_2(x,y,z) \\
\mathsf{G}(x,y) &= \exists z\, \mathsf{H}(x,y,z) \\
\mathsf{F}(x) &= \forall y\, \mathsf{G}(x,y).
\end{aligned}$$

Then

x	y	z	fx	fy	fz	$\mathsf{H}_1(x,y,z)$	$\mathsf{H}_2(x,y,z)$	$\mathsf{H}(x,y,z)$
a	a	a	b	b	b	0	0	0
a	a	b	b	b	a	0	1	0
a	b	a	b	a	b	1	1	1
a	b	b	b	a	a	1	0	0
b	a	a	a	b	b	1	0	0
b	a	b	a	b	a	1	1	1
b	b	a	a	a	b	0	1	0
b	b	b	a	a	a	0	0	0

x	y	$\mathsf{G}(x,y)$
a	a	0
a	b	1
b	a	1
b	b	0

x	$\mathsf{F}(x)$
a	0
b	0

$\mathbf{S} = \{a, b\}$

r	a	b
a	0	1
b	1	0

	f
a	b
b	a

$$\mathsf{F}(x, y) = \exists z\, ((rxfz) \wedge (fy \approx z)) \rightarrow (fy \approx fx).$$

$\boxed{\mathsf{F}^{\mathbf{S}} =?}$ Let

$$\begin{aligned}
\mathsf{G}_1(x, y, z) &= rxfz \\
\mathsf{G}_2(x, y, z) &= fy \approx z \\
\mathsf{G}(x, y, z) &= \mathsf{G}_1(x, y, z) \wedge \mathsf{G}_2(x, y, z) \\
\mathsf{F}_1(x, y) &= \exists z\, \mathsf{G}(x, y, z) \\
\mathsf{F}_2(x, y) &= fy \approx fx \\
\mathsf{F}(x, y) &= \mathsf{F}_1(x, y) \rightarrow \mathsf{F}_2(x, y).
\end{aligned}$$

Then

x	y	z	fx	fy	fz	$\mathsf{G}_1(x,y,z)$	$\mathsf{G}_2(x,y,z)$	$\mathsf{G}(x,y,z)$
a	a	a	b	b	b	1	0	0
a	a	b	b	b	a	0	1	0
a	b	a	b	a	b	1	1	1
a	b	b	b	a	a	0	0	0
b	a	a	a	b	b	0	0	0
b	a	b	a	b	a	1	1	1
b	b	a	a	a	b	0	1	0
b	b	b	a	a	a	1	0	0

so

x	y	$\mathsf{F}_1(x,y)$	$\mathsf{F}_2(x,y)$	$\mathsf{F}(x,y)$
a	a	0	1	1
a	b	1	0	0
b	a	1	0	0
b	b	0	1	1

Let F be a sentence and $\mathbf{S}$ a structure.

Then F is **true** in $\mathbf{S}$ provided one of the following holds:

- F is $\boxed{rt_1 \cdots t_n}$ and $r^{\mathbf{S}}(t_1^{\mathbf{S}}, \ldots, t_n^{\mathbf{S}})$ holds;
- F is $\boxed{t_1 \approx t_2}$ and $t_1^{\mathbf{S}} = t_2^{\mathbf{S}}$;
- F is $\boxed{\neg\, \mathsf{G}}$ and G is not true in $\mathbf{S}$;
- F is $\boxed{\mathsf{G} \vee \mathsf{H}}$ and at least one of G, H is true in $\mathbf{S}$;

- F is $\boxed{\mathsf{G} \wedge \mathsf{H}}$ and both of G, H are true in $\mathbf{S}$;

- F is $\boxed{\mathsf{G} \rightarrow \mathsf{H}}$ and G is not true in $\mathbf{S}$ or H is true in $\mathbf{S}$;

- F is $\boxed{\mathsf{G} \leftrightarrow \mathsf{H}}$ and both or neither of G, H is true in $\mathbf{S}$;

- F is $\boxed{\forall x\, \mathsf{G}(x)}$ and $\mathsf{G}^{\mathbf{S}}(a)$ is true for every $a \in \mathbf{S}$;

- F is $\boxed{\exists x\, \mathsf{G}(x)}$ and $\mathsf{G}^{\mathbf{S}}(a)$ is true for some $a \in \mathbf{S}$.

If F is not true in $\mathbf{S}$, then we say F is **false** in $\mathbf{S}$.

Given a first–order language $\mathcal{L}$, let F be a sentence, $\mathcal{S}$ a set of sentences, and $\mathbf{S}$ a structure for this language.

- $\boxed{\mathbf{S} \models \mathrm{F}}$ means F is true in $\mathbf{S}$.

- $\boxed{\mathrm{F}}$ is **valid** if it is true in all $\mathcal{L}$–structures.

- $\boxed{\mathbf{S} \models \mathcal{S}}$ means every sentence F in $\mathcal{S}$ is true in $\mathbf{S}$.

- $\boxed{\mathrm{Sat}(\mathcal{S})}$ means $\mathcal{S}$ is satisfiable.

- $\boxed{\mathcal{S} \models \mathrm{F}}$ means every model of $\mathcal{S}$ is a model of F. We say F **is a consequence of** $\mathcal{S}$.

The Propositional Skeleton

(A simple test for having a one–element model.)

The **propositional skeleton**, Skel(F), of a formula is defined as follows:

- Delete all quantifiers and terms.
- Replace $\approx$ with 1.
- Replace the relation symbols r with propositional variables R.

Theorem

F has a **one–element** model iff Skel(F) is satisfiable.

If Skel(F) is satisfiable, then choose an evaluation $\mathbf{e}$ that makes it true.

Let $r^{\mathbf{S}}(a, \ldots, a)$ hold iff $\mathbf{e}(R) = 1$ for $r \in \mathcal{R}$.

Let $f^{\mathbf{S}}(a, ..., a) = a$ for $f \in \mathcal{F}$.

To illustrate this, we consider the first–order sentence F given by

$$\forall x \forall y \, (\neg \, (x < y) \leftrightarrow \exists z \, ((x < z) \vee (f z \approx y))).$$

Then Skel(F) is the propositional formula

$$\neg \, P \leftrightarrow P \vee 1$$

This is satisfiable by evaluating P as 0, so F has a one–element model on the set $\{a\}$, namely

$$\begin{array}{c|c} < & a \\ \hline a & 0 \end{array}, \quad \text{and} \quad f a = a$$

Equivalent Sentences

The sentences F and G are **equivalent**, written $\boxed{\mathsf{F} \sim \mathsf{G}}$, if they are true of the same $\mathcal{L}$–structures $\mathbf{S}$,
that is, for all structures $\mathbf{S}$ we have

$$\mathbf{S} \models \mathsf{F} \quad \text{iff} \quad \mathbf{S} \models \mathsf{G}.$$

$$\forall x\,(\neg\,(x \approx 0) \to \exists y\,(x \cdot y \approx 1))$$

and

$$\forall x \exists y\,(\neg\,(x \approx 0) \to (x \cdot y \approx 1))$$

are equivalent.

Proposition

The sentences F and G are equivalent iff $\mathsf{F} \leftrightarrow \mathsf{G}$ is a valid sentence.

Equivalent Formulas

Two formulas $\mathsf{F}(x_1, \ldots, x_n)$ and $\mathsf{G}(x_1, \ldots, x_n)$ are **equivalent**, written

$$\boxed{\mathsf{F}(x_1, \ldots, x_n) \sim \mathsf{G}(x_1, \ldots, x_n)},$$

iff F and G define the same relation on any $\mathcal{L}$–structure $\mathbf{S}$, that is, $\mathsf{F}^{\mathbf{S}} = \mathsf{G}^{\mathbf{S}}$.

$$\neg (x \approx 0) \to \exists y \, (x \cdot y \approx 1)$$

and

$$\exists y \, (\neg (x \approx 0) \to (x \cdot y \approx 1))$$

are equivalent.

Proposition

The formulas $\mathsf{F}(\vec{x})$ and $\mathsf{G}(\vec{x})$ are equivalent iff $\forall\vec{x}(\mathsf{F}(\vec{x}) \leftrightarrow \mathsf{G}(\vec{x}))$ is a valid sentence.

Proposition

The relation $\sim$ is an equivalence relation on sentences as well as on formulas.

This is immediate from the definition of $\sim$ and the fact that ordinary equality ($=$) is an equivalence relation.

Fundamental Equivalences

$$\neg\,\exists x\,\mathsf{F} \;\sim\; \forall x\,(\neg\,\mathsf{F})$$

$$\neg\,\forall x\,\mathsf{F} \;\sim\; \exists x\,(\neg\,\mathsf{F})$$

$$(\forall x\,\mathsf{F}) \vee \mathsf{G} \;\sim\; \forall x\,(\mathsf{F} \vee \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$(\exists x\,\mathsf{F}) \vee \mathsf{G} \;\sim\; \exists x\,(\mathsf{F} \vee \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$(\forall x\,\mathsf{F}) \wedge \mathsf{G} \;\sim\; \forall x\,(\mathsf{F} \wedge \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$(\exists x\,\mathsf{F}) \wedge \mathsf{G} \;\sim\; \exists x\,(\mathsf{F} \wedge \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$(\forall x\,\mathsf{F}) \to \mathsf{G} \;\sim\; \exists x\,(\mathsf{F} \to \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$(\exists x\,\mathsf{F}) \to \mathsf{G} \;\sim\; \forall x\,(\mathsf{F} \to \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{G}$$

$$\mathsf{F} \to (\forall x\,\mathsf{G}) \;\sim\; \forall x\,(\mathsf{F} \to \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{F}$$

$$\mathsf{F} \to (\exists x\,\mathsf{G}) \;\sim\; \exists x\,(\mathsf{F} \to \mathsf{G}) \qquad \text{if } x \text{ is not free in } \mathsf{F}$$

$$\forall x\,(\mathsf{F} \wedge \mathsf{G}) \;\sim\; (\forall x\,\mathsf{F}) \wedge (\forall x\,\mathsf{G})$$

$$\exists x\,(\mathsf{F} \vee \mathsf{G}) \;\sim\; (\exists x\,\mathsf{F}) \vee (\exists x\,\mathsf{G})$$

Notes on the Fundamental Equivalences

The first two, combined with the fundamental laws from propositional logic, are very useful in practice, e.g., the statement

f is continuous at x:

$$\forall\varepsilon((\varepsilon > 0) \to \exists\delta((\delta > 0) \wedge \forall y(|x - y| < \delta \to |fx - fy| < \varepsilon))),$$

negates to

f is not continuous at x:

$$\exists\varepsilon((\varepsilon > 0) \wedge \forall\delta((\delta > 0) \to \exists y((|x - y| < \delta) \wedge \neg(|fx - fy| < \varepsilon))$$

Comment

If x occurs free in G then we cannot conclude

$$(\forall x\, \mathsf{F}) \vee \mathsf{G} \;\sim\; \forall x\, (\mathsf{F} \vee \mathsf{G}),$$

for example,

$(\forall x(x < 0)) \vee (0 < x)$ and $\forall x((x < 0) \vee (0 < x))$

are not equivalent, for in the natural numbers the first is true of positive numbers x, whereas the second is false in the natural numbers.

For implication we have:

$$\begin{aligned}(\forall x\,\mathsf{F}) \to \mathsf{G} &\sim \neg\,(\forall x\,\mathsf{F}) \vee \mathsf{G}\\ &\sim \exists x\,(\neg\,\mathsf{F}) \vee \mathsf{G}\\ &\sim \exists x\,(\neg\,\mathsf{F} \vee \mathsf{G})\\ &\sim \exists x\,(\mathsf{F} \to \mathsf{G})\end{aligned}$$

To see that $\forall x\,(\mathsf{F} \vee \mathsf{G}) \sim (\forall x\,\mathsf{F}) \vee (\forall x\,\mathsf{G})$ need not be true consider the following example:

$\forall x\,((0 \approx x) \vee (0 < x))$ and $(\forall x\,(0 \approx x)) \vee (\forall x\,(0 < x))$

And to see that $\exists x\,(\mathsf{F} \wedge \mathsf{G}) \sim (\exists x\,\mathsf{F}) \wedge (\exists x\,\mathsf{G})$ need not be true consider the example:

$\exists x\,((0 \approx x) \wedge (0 < x))$ and $(\exists x\,(0 \approx x)) \wedge (\exists x\,(0 < x))$

Replacement and Substitution

Equivalent propositional formulas lead to equivalent first–order formulas as follows.

If $\mathsf{F}(P_1, \ldots, P_n)$ and $\mathsf{G}(P_1, \ldots, P_n)$ are equivalent propositional formulas, then for any sequence $\mathsf{H}_1, \ldots, \mathsf{H}_n$ of first–order formulas we have

$$\mathsf{F}(\mathsf{H}_1, \ldots, \mathsf{H}_n) \sim \mathsf{G}(\mathsf{H}_1, \ldots, \mathsf{H}_n)$$

Example $\neg\,(P \wedge Q) \sim \neg\, P \vee \neg\, Q$ holds, so we have, in first–order logic,

$$\neg\,\Big((\exists x(x \cdot x \approx 1)) \wedge (\forall x \forall y(x \cdot y \approx y \cdot x))\Big)$$

$$\sim \quad \neg\,\Big(\exists x(x \cdot x \approx 1)\Big) \vee \neg\,\Big(\forall x \forall y(x \cdot y \approx y \cdot x)\Big)$$

Equivalence is compatible with our connectives and quantifiers in the obvious sense.

Lemma

Suppose $\mathsf{F}_1 \sim \mathsf{G}_1$ and $\mathsf{F}_2 \sim \mathsf{G}_2$. Then

$$\begin{aligned}
\neg\,\mathsf{F}_1 &\sim \neg\,\mathsf{G}_1 \\
\mathsf{F}_1 \vee \mathsf{F}_2 &\sim \mathsf{G}_1 \vee \mathsf{G}_2 \\
\mathsf{F}_1 \wedge \mathsf{F}_2 &\sim \mathsf{G}_1 \wedge \mathsf{G}_2 \\
\mathsf{F}_1 \to \mathsf{F}_2 &\sim \mathsf{G}_1 \to \mathsf{G}_2 \\
\mathsf{F}_1 \leftrightarrow \mathsf{F}_2 &\sim \mathsf{G}_1 \leftrightarrow \mathsf{G}_2 \\
\forall x\,\mathsf{F}_1 &\sim \forall x\,\mathsf{G}_1 \\
\exists x\,\mathsf{F}_1 &\sim \exists x\,\mathsf{G}_1.
\end{aligned}$$

This leads us to the basic replacement theorem for first–order logic.

Replacement Theorem

If $\mathsf{F} \sim \mathsf{G}$ then $\mathsf{H}(\cdots \mathsf{F} \cdots) \sim \mathsf{H}(\cdots \mathsf{G} \cdots)$.

Example

Since

$$\neg \Big((\exists x (x \cdot x \approx 1)) \ \wedge \ (\forall x \forall y (x \cdot y \approx y \cdot x)) \Big)$$

$$\sim \quad \neg \Big(\exists x (x \cdot x \approx 1) \Big) \ \vee \ \neg \Big(\forall x \forall y (x \cdot y \approx y \cdot x) \Big)$$

we have

$$(\forall x \exists y (x < y)) \rightarrow \neg \Big((\exists x (x \cdot x \approx 1)) \wedge (\forall x \forall y (x \cdot y \approx y \cdot x)) \Big)$$

$$\sim \ (\forall x \exists y (x < y)) \rightarrow \neg \Big(\exists x (x \cdot x \approx 1) \Big) \vee \neg \Big(\forall x \forall y (x \cdot y \approx y \cdot x) \Big)$$

Substitution in first-order logic often requires the need to rename variables.

We need to be careful with renaming variables to avoid binding any newly introduced occurrences of variables.

Given a first–order formula F, define a **conjugate** of F to be any formula $\overline{F}$ obtained by renaming the occurrences of bound variables of F so that no free occurrences of variables in F become bound. Keep bound occurrences of distinct variables distinct.

Conjugates are equivalent:

If $\overline{\mathsf{F}}$ is a conjugate of F, then $\overline{\mathsf{F}} \sim \mathsf{F}$.

Thus $\exists y(x \cdot y \approx 1) \sim \exists w(x \cdot w \approx 1)$.

Substitution Theorem
If

$$\mathsf{F}(x_1, \ldots, x_n) \sim \mathsf{G}(x_1, \ldots, x_n)$$

and $t_1, \ldots, t_n$ are terms, then

$$\mathsf{F}^\star(t_1, \ldots, t_n) \sim \mathsf{G}^\star(t_1, \ldots, t_n).$$

Thus from

$$\neg\, \exists y(x \cdot y \approx 1) \sim \forall y(\neg(x \cdot y \approx 1))$$

follows

$$\neg\, \exists u((y + w) \cdot u \approx 1) \sim \forall u(\neg((y + w) \cdot u \approx 1))$$

Prenex Form

A formula F is in **prenex** form if it looks like

$$Q_1 x_1 \cdots Q_n x_n \, \mathsf{G}$$

where the Q_i are quantifiers and G has no occurrences of quantifiers.

A formula with no occurrences of quantifiers is called an **open** formula.

The formula

$$\exists x \Big((rxy) \to \forall u \, (ruy) \Big)$$

is not in prenex form, but it is equivalent to the prenex form formula

$$\exists x \forall u \Big((rxy) \to (ruy) \Big).$$

Theorem

Every formula is equivalent to a formula in prenex form.

The following steps put F in prenex form:

- Rename the quantified variables so that

  distinct occurrences of quantifiers bind distinct variables, and

  no free variable is equal to a bound variable

Thus change

$$\forall z((rzy) \rightarrow \neg\, \forall y((rxy) \wedge \exists y(ryx)))$$

to

$$\forall z((rzy) \rightarrow \neg\, \forall u((rxu) \wedge \exists w(rwx)))$$

• Eliminate all occurrences of $\rightarrow$ and $\leftrightarrow$ using

$$\begin{array}{lcl} \mathsf{G} \leftrightarrow \mathsf{H} & \sim & (\neg\, \mathsf{G} \vee \mathsf{H}) \wedge (\neg\, \mathsf{H} \vee \mathsf{G}) \\ \mathsf{G} \rightarrow \mathsf{H} & \sim & \neg\, \mathsf{G} \vee \mathsf{H} \end{array}$$

The above example becomes

$$\forall z(\neg\, (rzy) \vee \neg\, \forall u((rxu) \wedge \exists w(rwx)))$$

• Pull the quantifiers to the front by repeated use of the following

$$
\begin{array}{rcl}
\neg(\mathsf{F} \vee \mathsf{G}) & \sim & (\neg \mathsf{F} \wedge \neg \mathsf{G}) \\
\neg(\mathsf{F} \wedge \mathsf{G}) & \sim & (\neg \mathsf{F} \vee \neg \mathsf{G}) \\
\mathsf{G} \vee (\forall x\, \mathsf{H}) & \sim & \forall x\, (\mathsf{G} \vee \mathsf{H}) \\
\mathsf{G} \vee (\exists x\, \mathsf{H}) & \sim & \exists x\, (\mathsf{G} \vee \mathsf{H}) \\
\mathsf{G} \wedge (\forall x\, \mathsf{H}) & \sim & \forall x\, (\mathsf{G} \wedge \mathsf{H}) \\
\mathsf{G} \wedge (\exists x\, \mathsf{H}) & \sim & \exists x\, (\mathsf{G} \wedge \mathsf{H}) \\
(\forall x\, \mathsf{G}) \vee \mathsf{H} & \sim & \forall x\, (\mathsf{G} \vee \mathsf{H}) \\
(\exists x\, \mathsf{G}) \vee \mathsf{H} & \sim & \exists x\, (\mathsf{G} \vee \mathsf{H}) \\
(\forall x\, \mathsf{G}) \wedge \mathsf{H} & \sim & \forall x\, (\mathsf{G} \wedge \mathsf{H}) \\
(\exists x\, \mathsf{G}) \wedge \mathsf{H} & \sim & \exists x\, (\mathsf{G} \wedge \mathsf{H}) \\
\neg \exists x\, \mathsf{G} & \sim & \forall x \neg \mathsf{G} \\
\neg \forall x\, \mathsf{G} & \sim & \exists x \neg \mathsf{G}
\end{array}
$$

Applying these steps to

$$\forall z(\neg\,(rzy) \vee \neg\,\forall u((rxu) \wedge \exists w(rwx)))$$

gives

$\forall z(\neg\,(rzy) \vee \neg\,\forall u((rxu) \wedge \exists w(rwx)))$
$\downarrow$
$\forall z(\neg\,(rzy) \vee \exists u(\neg\,((rxu) \wedge \exists w(rwx))))$
$\downarrow$
$\forall z\exists u(\neg\,(rzy) \vee \neg\,((rxu) \wedge \exists w(rwx)))$
$\downarrow$
$\forall z\exists u(\neg\,(rzy) \vee (\neg\,(rxu) \vee \neg(\exists w(rwx))))$
$\downarrow$
$\forall z\exists u(\neg\,(rzy) \vee (\neg\,(rxu) \vee \forall w(\neg(rwx))))$
$\downarrow$
$\forall z\exists u(\neg\,(rzy) \vee \forall w(\neg\,(rxu) \vee (\neg(rwx))))$
$\downarrow$
$\forall z\exists u\forall w(\neg\,(rzy) \vee (\neg\,(rxu) \vee (\neg(rwx))))$

Valid Arguments

Now we are working with **sentences** in a fixed first–order language $\mathcal{L}$.

An argument $\mathsf{F}_1, \cdots, \mathsf{F}_n \ \therefore \mathsf{F}$ is **valid** (or **correct**) in first–order logic provided every structure $\mathbf{S}$ that makes $\mathsf{F}_1, \ldots, \mathsf{F}_n$ true also makes F true, that is, for all $\mathbf{S}$

$$\mathbf{S} \models \{\mathsf{F}_1, \ldots, \mathsf{F}_n\} \quad \text{implies} \quad \mathbf{S} \models \mathsf{F}.$$

Proposition

An argument

$$\mathsf{F}_1, \cdots, \mathsf{F}_n \ \therefore \mathsf{F}$$

in first–order logic is valid iff

$$\mathsf{F}_1 \wedge \cdots \wedge \mathsf{F}_n \rightarrow \mathsf{F}$$

is a valid sentence; and this holds iff

$$\{\mathsf{F}_1, \ \cdots \ \mathsf{F}_n, \ \neg \mathsf{F}\}$$

is not satisfiable.

Example

The work in Chapter 3 concerned equations. In first-order logic we treat equations as universally quantified sentences:

$$\forall \vec{x}(s(\vec{x}) \approx t(\vec{x}))$$

The argument

$$\forall x \forall y \forall u \forall v \, (x \cdot y \approx u \cdot v)$$
$$\therefore \forall x \forall y \forall z \, ((x \cdot y) \cdot z \approx x \cdot (y \cdot z))$$

is valid.

This is evident by considering that if a structure $\mathbf{S}$ satisfies the premiss then all multiplications give the same value. But then the multiplication must be associative.

The argument $\begin{array}{l}\exists y \forall x\,(rxy) \\ \therefore\, \forall x \exists y\,(rxy)\end{array}$ is valid.

To see this suppose $\mathbf{S}$ is a structure satisfying the premiss. Then choose an element a from S such that $\forall x\,(rxa)$ holds. From this we can conclude that $\forall x \exists y\,(rxy)$ also holds since we can let y be a.

We have demonstrated the validity of the above arguments by appealing to our reasoning skills in mathematics.

It is fair to ask if one can now sit down and write out all the axioms and rules that one would ever need to justify valid arguments in first-order logic.

This question was posed by **Hilbert and Ackerman** in 1928 in their slender book, the elementary and famous **Principles of Logic**.

They proposed a collection of first-order axioms and rules of inference, and noted that they were correct rules of reasoning.

But were they enough, that is, was their system **complete**?

They said in their book that their proof system was sufficient for doing all the things they tried with first-order logic, and indeed asked if it was complete.

The brilliant young student **Kurt Gödel** in Vienna answered their question within a year, in the affirmitive, and used it for his PhD Thesis.

This is the famous **Gödel Completeness Theorem**, discussed in Chapter 6 of the text. It says

$$\mathcal{S} \models \varphi \quad \text{iff} \quad \mathcal{S} \vdash \varphi.$$

Another question that was prominent at the time was whether or not the powerful logical system of **Principia Mathematica**, developed by **Whitehead and Russell** in 3 volumes (1910-1913), was powerful enough to do all mathematics, forever.

This system of logic was **higher-order**, meaning that one could quantify over things other than just the variables for the elements of the universe. One could also quantify over relation symbols (and function symbols).

Thus they could write $\forall r(rx \leftrightarrow ry)$ as a **definition** of $x \approx y$.

It was known, from looking at the examples in Principia, that all known mathematical proofs could be written up in this system (if one so desired).

So perhaps this is all one would ever need.

Within a year of his PhD Gödel had answered this (much more difficult) question as well.

To everyones astonishment he put together a (lowly) **first-order** sentence F in the language of the natural numbers $\mathbf{N} = \{N, +, \cdot, <, 0, 1\}$ that he could show was

- True of $\mathbf{N}$

- But Principia Mathematica was not strong enough to prove it.

(Assuming that the proof system of Principia did not lead to a contradiction $\mathsf{G} \wedge \neg \mathsf{G}$, for then one could derive every formula!)

This is the famous **Gödel Incompleteness Theorem**.

Counterexamples

To show that an argument $\mathsf{F}_1, \cdots, \mathsf{F}_n \ \therefore \mathsf{F}$ is not valid it suffices to find a single structure $\mathbf{S}$ such that

- each of the premisses $\mathsf{F}_1, \cdots, \mathsf{F}_n$ is true in $\mathbf{S}$, but

- the conclusion F is false in $\mathbf{S}$.

Such a structure $\mathbf{S}$ is a **counterexample** to the argument.

The argument $\begin{array}{l} \forall x \exists y\,(rxy) \\ \therefore \exists y \forall x\,(rxy) \end{array}$ is not valid; a two–element graph gives a counterexample:

Skolemization

Skolem, following the work of Löwenheim (1915), gave us a technique to convert a first-order sentence F into a sentence F' in prenex form, with only universal quantifiers, such that

F is satisfiable iff F' is satisfiable.

Universally quantified sentences are apparently much easier to understand.

This has provided one of the powerful techniques in automated theorem proving.

• One takes a given assertion in mathematics and expresses it as a first-order sentence F.

This means we want the assertion to be true iff F is valid (i.e., true in **all** structures).

• This sentence F is valid iff $\neg F$ is not satisfiable.

• One replaces $\neg F$ by a universal sentence F' obtained by Skolem's process.

• Then F is valid iff F' is not satisfiable.

• One puts the matrix of F' in conjunctive form.

• Then it is easy to replace F' by a finite set of clauses $\mathcal{C}'$ such that F' is not satisfiable iff $\mathcal{C}'$ is not satisfiable.

Namely the clauses come from the conjuncts of the matrix of F'.

• Now one has F is valid iff $\mathcal{C}'$ is not satisfiable.

To show $\mathcal{C}'$ is not satisfiable one can apply resolution theorem proving.

Now we look at the details of Skolem's process, called skolemization.

Lemma

- Given the sentence $\exists y\, \mathsf{G}(y)$,

augment the language with a new constant c and form the sentence $\mathsf{G}(c)$. Then

$$\mathsf{Sat}\Big(\exists y\, \mathsf{G}(y)\Big) \quad \text{iff} \quad \mathsf{Sat}\Big(\mathsf{G}(c)\Big).$$

- Given the sentence $\forall x_1 \cdots \forall x_n \exists y\, \mathsf{G}(\vec{x}, y)$,

augment the language with a new n–ary function symbol f and form the sentence $\forall x_1 \cdots \forall x_n\, \mathsf{G}^\star(\vec{x}, f(\vec{x}))$. Then

$$\mathsf{Sat}\Big(\ \forall \vec{x}\, \exists y\, \mathsf{G}(\vec{x}, y)\ \Big) \quad \text{iff} \quad \mathsf{Sat}\Big(\ \forall \vec{x}\, \mathsf{G}^\star(\vec{x}, f(\vec{x}))\ \Big).$$

Proof

In each step we can expand a model of the sentence on the left–hand side to one on the right–hand side by adding the appropriately designated element for the constant, or by adding an appropriate function f.

Conversely, we can take a model of the sentence on the right–hand side and remove the constant symbol, respectively function symbol, and get a model for the left–hand side.

Definition

A first–order formula F is **universal** if it is in prenex form and all quantifiers are universal,

that is, F is of the form $\boxed{\forall \vec{x}\, \mathsf{G}}$, where G is quantifier–free.

G is called the **matrix** of F.

Example

$$\forall x \forall y \forall z \underbrace{((x \leq y) \wedge (y \leq z) \rightarrow (x \leq z))}_{\text{matrix}}$$

Theorem

Given a first–order sentence F, there is an effective procedure for finding a universal sentence F' (usually in an extended language) such that

$$\mathrm{Sat}(F) \quad \text{iff} \quad \mathrm{Sat}(F').$$

Furthermore, we can choose F' such that every model of F can be expanded to a model of F', and every model of F' can be reducted to a model of F.

Proof

First, we put F in prenex form. Then we just apply the Lemma repeatedly until there are no existential quantifiers.

The process of converting a sentence to such a universal sentence is called **skolemizing**.

The new constants and functions are called **skolem constants** and **skolem functions**.

Example

To skolemize the sentence

$$\mathsf{F} \;=\; \forall x \forall y \left((x < y) \rightarrow \exists z \left((x < z) \wedge (z < y) \right) \right)$$

we first put it in prenex form

$$\mathsf{F} \;=\; \forall x \forall y \exists z \left((x < y) \rightarrow (x < z) \wedge (z < y) \right)$$

Applying the Lemma, we introduce a new binary function symbol, say f, and arrive at the universal sentence

$$\mathsf{F}' \;=\; \forall x \forall y \, ((x < y) \to (x < f(x,y)) \wedge (f(x,y) < y))$$

The structure $\mathbf{Q} = (Q, <)$, consisting of the real numbers with the usual $<$, satisfies F.

If we choose $f(a,b) = \frac{a+b}{2}$ for $a, b \in Q$, we see that the expansion $(Q, <, f)$ of $\mathbf{Q}$ satisfies F′.

Now we extend these ideas to sets of sentences.

Theorem

Given a set of first–order sentences $\mathcal{S}$, there is a set $\mathcal{S}'$ of universal sentences (usually in an extended language) such that

$$\text{Sat}(\mathcal{S}) \quad \text{iff} \quad \text{Sat}(\mathcal{S}').$$

Furthermore, every model of $\mathcal{S}$ can be expanded to a model of $\mathcal{S}'$, and every model of $\mathcal{S}'$ can be reducted to a model of $\mathcal{S}$.

Proof

We skolemize each sentence in $\mathcal{S}$ as before, making sure that distinct sentences do not have any common skolem constants or functions.

This theorem is at the heart of the translation from first–order logic to clause logic, as we shall see in the next section.

Example

We skolemize the set of sentences

$$\{\exists x \forall y \exists z\,(x < y + z),\ \exists x \forall y \exists z\,(\neg\,(x < y + z))\}$$

and obtain a set of universal sentences

$$\{\forall y\,(a < y + fy),\ \forall x\,(\neg\,(b < y + gy))\}.$$

The Reduction of First-Order Logic to Clause Logic

Now we show that first–order formulas can be reduced to the predicate clause logic, and thus to the propositional clause logic.

First we want to look at some basic translations between clauses and first–order formulas. A clause

$$\mathsf{C} \;=\; \{\mathsf{L}_1(\vec{x}), \ldots, \mathsf{L}_k(\vec{x})\}$$

is considered as the universal sentence

$$\mathsf{F}_\mathsf{C} \;=\; \forall \vec{x}\,(\mathsf{L}_1(\vec{x}) \vee \cdots \vee \mathsf{L}_k(\vec{x}))$$

in first-order logic.

For indeed

$$\mathbf{S} \models \mathsf{C} \quad \text{iff} \quad \mathbf{S} \models \mathsf{F}_{\mathsf{C}}.$$

There is also a translation from universal first–order sentences to clauses.

Lemma

One can effectively construct, for any given universal sentence F, a finite set $\mathcal{C}_{\mathsf{F}}$ of clauses such that for any structure $\mathbf{S}$

$$\mathbf{S} \models \mathsf{F} \quad \text{iff} \quad \mathbf{S} \models \mathcal{C}_{\mathsf{F}} \ .$$

Proof

Let F be $\forall\vec{x}\,\mathsf{G}(\vec{x})$ with $\mathsf{G}(\vec{x})$ quantifier–free. Put $\mathsf{G}(\vec{x})$ in conjunctive form

$$\mathsf{G}(\vec{x}) \;\sim\; \mathsf{G}_1(\vec{x}) \wedge \cdots \wedge \mathsf{G}_k(\vec{x}),$$

where each $\mathsf{G}_i(\vec{x})$ is a disjunction of literals. Then

$$\begin{aligned}\forall\vec{x}\,\mathsf{G}(\vec{x}) \;&\sim\; \forall\vec{x}\,(\mathsf{G}_1(\vec{x}) \wedge \cdots \wedge \mathsf{G}_k(\vec{x}))\\ &\sim\; \mathsf{F}_1 \wedge \cdots \wedge \mathsf{F}_k\end{aligned}$$

where $\mathsf{F}_i \;=\; \forall\vec{x}\,\mathsf{G}_i(\vec{x})$. Thus

$$\mathbf{S} \models \mathsf{F} \quad \text{iff} \quad \mathbf{S} \models \{\mathsf{F}_1, \cdots, \mathsf{F}_k\}.$$

Let C_i be the clause $\mathsf{C}_{\mathsf{F}_i}$ associated with the matrix of F_i. Then

$$\mathbf{S} \models \mathsf{F} \quad \text{iff} \quad \mathbf{S} \models \mathcal{C}_{\mathsf{F}}$$

Example

Let

$$\mathsf{F} \;=\; \forall x \forall y \forall z \, (rx \;\rightarrow\; ry \wedge rz).$$

First, we put the matrix of F in conjunctive form:

$$\begin{aligned} rx \;\rightarrow\; ry \wedge rz \;\; &\sim \;\; \neg\, rx \;\vee\; (ry \wedge rz) \\ &\sim \;\; (\neg\, rx \;\vee\; ry) \;\wedge\; (\neg\, rx \;\vee\; rz). \end{aligned}$$

Now we can read off the clauses from the conjuncts, namely,

$$\mathcal{C}_{\mathsf{F}} \;=\; \{\; \{\neg\, rx,\; ry\},\; \{\neg\, rx,\; rz\} \;\}\,.$$

Definition For $\mathcal{S}$ a set of universal sentences let

$$\mathcal{C}_{\mathcal{S}} = \bigcup_{\mathsf{F} \in \mathcal{S}} \mathcal{C}_{\mathsf{F}},$$

the union of the collection of all sets $\mathcal{C}_{\mathsf{F}}$ of clauses obtained from the universal sentences F in $\mathcal{S}$.

Lemma Given a set $\mathcal{S}$ of universal sentences and $\mathbf{S}$ a structure,

$$\mathbf{S} \models \mathcal{S} \quad \text{iff} \quad \mathbf{S} \models \mathcal{C}_{\mathcal{S}}.$$

Thus

$$\mathsf{Sat}(\mathcal{S}) \quad \text{iff} \quad \mathsf{Sat}(\mathcal{C}_{\mathcal{S}}).$$

Now we can formulate a connection between satisfiability of a set of first–order sentences (not necessarily universal) and a set of clauses.

Lemma

Given a set $\mathcal{S}$ of first–order sentences, let $\mathcal{S}'$ be a skolemization of $\mathcal{S}$. Then

$$\mathrm{Sat}(\mathcal{S}) \quad \text{iff} \quad \mathrm{Sat}(\mathcal{C}_{\mathcal{S}'}).$$

Example

For $\mathcal{S} = \{\exists x \forall y\,(rxy),\ \neg\,\forall y \exists x\,(rxy)\}$ we have

$$\begin{aligned} \mathcal{S}' &= \{\forall y\,(ray),\ \forall x\,\neg\,(rxb)\} \\ \mathcal{C}'_{\mathcal{S}} &= \{\{ray\},\ \{\neg\, rxb\}\}. \end{aligned}$$

We can easily derive the empty clause from $\mathcal{C}'_{\mathcal{S}}$ (in one resolution step), so $\mathcal{C}'_{\mathcal{S}}$ is not satisfiable. Thus $\mathcal{S}$ is also not satisfiable.

Consequently, the sentence

$$\mathsf{F} = (\exists x \forall y\,(rxy)) \wedge (\neg\,\forall y \exists x\,(rxy)),$$

obtained by conjuncting the members of $\mathcal{S}$, is not satisfiable.

So $\neg\,\mathsf{F}$ is a valid sentence. Clearly

$$\neg\,\mathsf{F} \sim (\exists x \forall y\,(rxy)) \rightarrow (\forall y \exists x\,(rxy))$$

Now we can state the fundamental transformation from first–order logic to clause logic.

Theorem

Given $\mathcal{S}$, a set of sentences, and F, a sentence,

$$\mathcal{S} \models \mathsf{F} \quad \text{iff} \quad \neg\, \mathsf{Sat}(\ \mathsf{C}_{\mathcal{S}(\neg\,\mathsf{F})'}\ ),$$

where $\mathcal{S}(\neg\,\mathsf{F})$ is the set $\mathcal{S} \cup \{\neg\,\mathsf{F}\}$, and the prime symbol refers to taking the skolemization, as before.

Thus the argument $\boxed{\mathcal{S} \ \therefore \mathsf{F}}$ is valid iff one cannot satisfy the set $\mathsf{C}_{\mathcal{S}(\neg\,\mathsf{F})'}$ of clauses.

Example

$$\forall x \forall y \Big((rxy) \to \neg\, (ryx) \Big)$$
$$\forall x \forall y \forall z \Big(\Big((rxy) \wedge (ryz) \Big) \to (rxz) \Big)$$
$$\therefore\ \forall x \forall y\, (rxy \to \neg\, (x \approx y)).$$

Let F be the conclusion. Then

$$\neg\, \mathsf{F} \ \sim\ \exists x \exists y\, (rxy \wedge (x \approx y)).$$

The premisses $\mathcal{S}$ are already skolemized, as they are universal sentences. For $\neg\, \mathsf{F}$ we obtain the skolemized form

$$(rab) \wedge (a \approx b).$$

Thus the set of clauses corresponding to the argument is

$$\{\neg\, rxy,\ \neg\, ryx\}$$
$$\{\neg\, rxy,\ \neg\, ryz,\ rxz\}$$
$$\{rab\},\ \{a \approx b\}$$

We can also handle equational arguments.

Example

Consider the argument

$$\forall x\,(x \cdot 1 \;\approx\; x)$$
$$\forall x \exists y\,(x \cdot y \;\approx\; 1)$$
$$\forall x \forall y \forall z\,\Big((x \cdot y) \cdot z \;\approx\; x \cdot (y \cdot z)\Big)$$
$$\therefore \forall x\,(1 \cdot x \;\approx\; x).$$

The negation of the conclusion is the formula $\exists x\,(\neg\,(1 \cdot x \;\approx\; x))$.

Thus skolemizing the premisses with the negated conclusion gives

$$\forall x\,(x \cdot 1 \;\approx\; x)$$
$$\forall x\,(x \cdot fx \;\approx\; 1)$$
$$\forall x \forall y \forall z\,\Big((x \cdot y) \cdot z \;\approx\; x \cdot (y \cdot z)\Big)$$
$$\neg\,(1 \cdot a \;\approx\; a)$$

These are ready to be converted to clauses, giving

$$\{x \cdot 1 \;\approx\; x\}$$
$$\{x \cdot fx \;\approx\; 1\}$$
$$\{(x \cdot y) \cdot z \;\approx\; x \cdot (y \cdot z)\}$$
$$\{\neg\,(1 \cdot a \;\approx\; a)\}$$

We can convert

$$\{x \cdot 1 \approx x\} \qquad \{(x \cdot y) \cdot z \approx x \cdot (y \cdot z)\}$$
$$\{x \cdot fx \approx 1\} \qquad \{\neg(1 \cdot a \approx a)\}$$

into a set of clauses without equality:

$$\{x \equiv x\}$$
$$\{x \not\equiv y,\ y \equiv x\}$$
$$\{x \not\equiv y,\ y \not\equiv z,\ x \equiv z\}$$
$$\{x \not\equiv y,\ fx \equiv fy\}$$
$$\{x_1 \not\equiv y_1,\ x_2 \not\equiv y_2,\ x_1 \cdot x_2 \equiv y_1 \cdot y_2\}$$
$$\{x \cdot 1 \equiv x\}$$
$$\{x \cdot fx \equiv 1\}$$
$$\{(x \cdot y) \cdot z \equiv x \cdot (y \cdot z)\}$$
$$\{1 \cdot a \not\equiv a\}$$

Now we are ready to apply resolution theorem proving. (Go to Chapter 4!)