Math Logic

Meaning of symbols used here:

P, Q, other upper case letters: propositions - are true or false

$Q \rightarrow P$: if Q is true then P is true, name of $\rightarrow$ is 'implies'

$\neg Q$: true if Q is false and false if Q is true, called 'complement'

$Q \vee P$: true if either Q is true or P is true, called inclusive 'or'

$Q \wedge P$: true only if both Q and P are true, called 'and'

$Q \oplus P$: true if one of Q or P is true but not both, exclusive 'or'

$Q \leftrightarrow P$: Q is true if and only if P is true, called 'iff'

$Q \equiv P$: Q is equivalent to P, always true

$P(x,...)$: a predicate – value depends on x

$\exists\, x, P(x,\ldots)$: there is a value for x such that $P(x,\ldots)$ is true

$\forall x, P(x,\ldots)$: for all x $P(x,\ldots)$ is true

iff: if and only if

What is Math Logic?

Set of mathematical disciplines including Boolean algebra, predicate calculus, propositional calculus, set theory, model theory, recursion theory, and proof theory with the aim of reducing formal logic to algebra

What is Math Logic?

Set of mathematical disciplines including Boolean algebra, predicate calculus, propositional calculus, set theory, model theory, recursion theory, and proof theory with the aim of reducing formal logic to algebra

Why do we want to study Math Logic?

Three reasons:

1. Logic is needed to reason about the behavior of hardware and software – to help determine whether hardware or software developed by “good guys” has any vulnerabilities and to determine whether unknown hardware or software is malicious
2. If logic manipulations can be reduced to algebra it becomes possible to build mechanical systems to prove theorems symbolically (no testing).
3. People often make mistakes in logic

Why study Math Logic?

Example: logic manipulations are reduced to algebra:

Let Q = constraints that represent the operation of a circuit or program snipit.

P = constraints that represent a property that we would like to show holds for the circuit …

We want to show $Q \rightarrow P$
which is the same logically as $\neg Q \lor P$

Proving this is the same as proving $\neg(\neg Q \lor P)$ false
This is the same as proving $Q \land \neg P$ false

See also https://www.math.fsu.edu/~pkirby/mad2104/SlideShow/s2_2.pdf

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently (false premise)
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

But “if the streets are wet, it has rained recently” is a false premise because the streets may be wet for other reasons such as a street cleaner just exploded and spayed water all over the place. Hence one *cannot* conclude that it has rained recently.

Note: if the premise “the streets are not wet” is used instead then one can conclude it is not raining!

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently (false premise)
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

Incomplete Premise:

(Does not cover facts necessary to prove a conclusion)
hunters want a good natural environment (premise)
therefore hunters are environmentalists (conclusion)

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently (false premise)
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

Incomplete Premise:

(Does not cover facts necessary to prove a conclusion)
hunters want a good natural environment (premise)
therefore hunters are environmentalists (conclusion)

person wanting a good natural environment is
an environmentalist (missing)

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently (false premise)
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

Incomplete Premise:

(Does not cover facts necessary to prove a conclusion)
hunters want a good natural environment (premise)
therefore hunters are environmentalists (conclusion)

person wanting a good natural environment is
an environmentalist (missing)

Hidden Premise:

greenhouse gas emissions must be regulated to avoid
dangerous manmade interference with the climate system

Problems of Human Mistakes in Logic:

Wrong (false) Premise:

if the streets are wet, it has rained recently (false premise)
the streets are wet (premise)
therefore it has rained recently (conclusion)

(If $A \rightarrow B$; and if A ; then infer B) is perfectly valid!!

Incomplete Premise:

(Does not cover facts necessary to prove a conclusion)
hunters want a good natural environment (premise)
therefore hunters are environmentalists (conclusion)

person wanting a good natural environment is
an environmentalist (missing)

Hidden Premise:

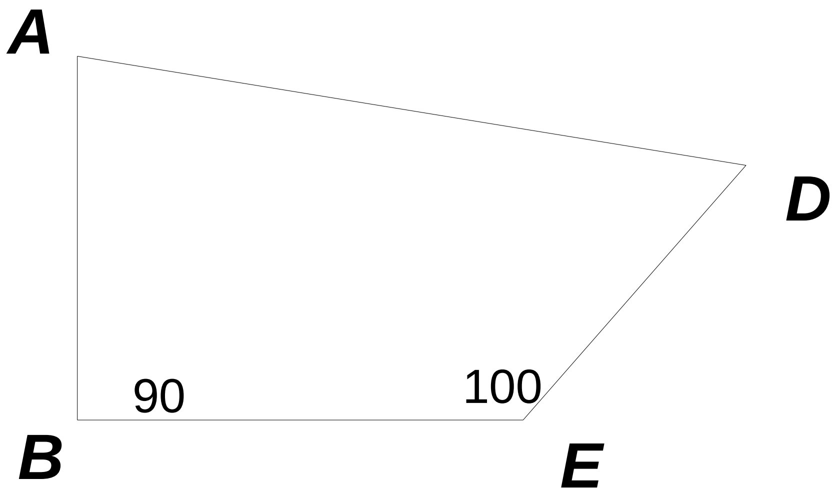
greenhouse gas emissions must be regulated to avoid
dangerous manmade interference with the climate system

dangerous human activity can interfere with climate (hidden)

Theorem: 90=100

Theorem: 90=100

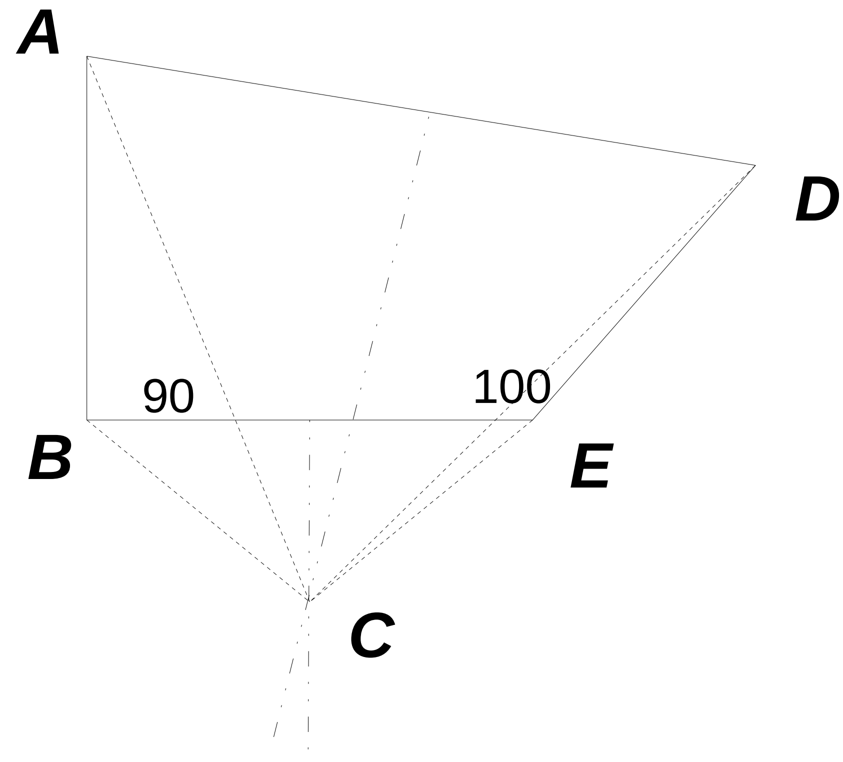
Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Theorem: 90=100

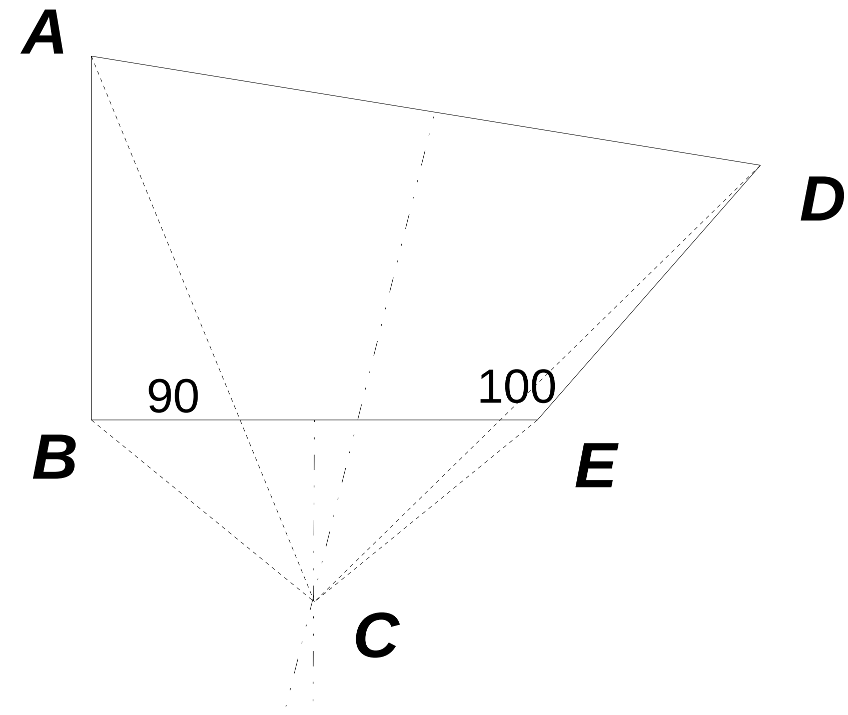
Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$

Draw perpendicular bisectors to *BE* and *AD* - they meet at point *C*



Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



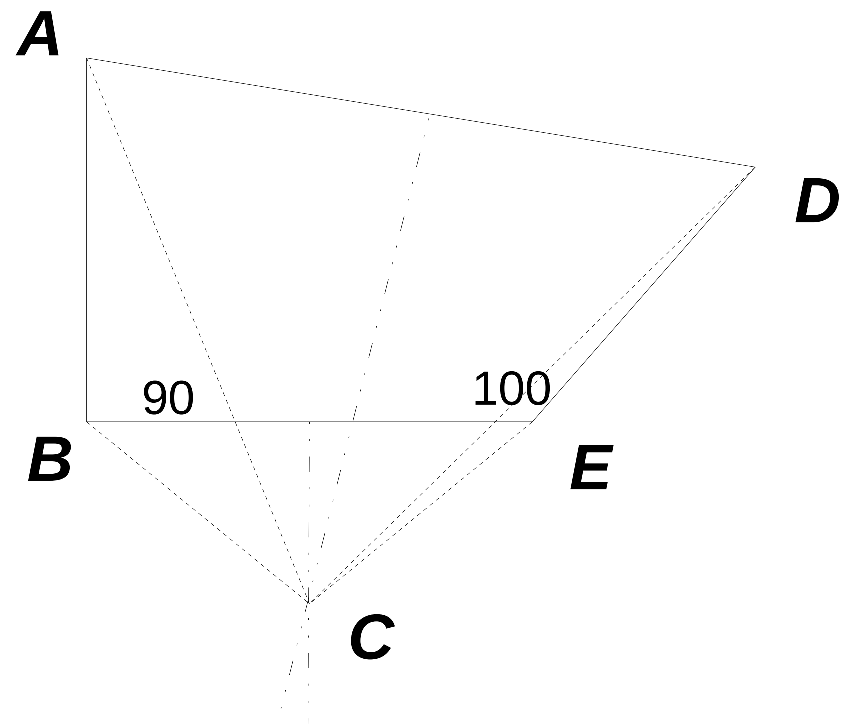
Draw perpendicular bisectors to *BE* and *AD* - they meet at point *C*

$|BC| = |EC|$ - *C* is on the perpendicular bisector of *BE*

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$

Draw perpendicular bisectors to *BE* and *AD* - they meet at point *C*



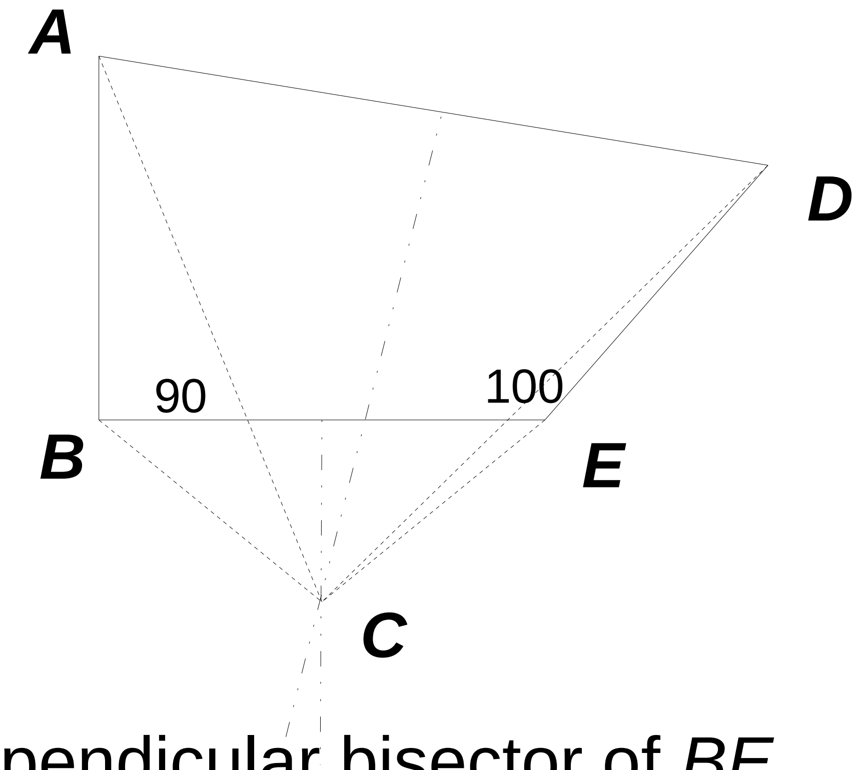
$|BC| = |EC|$ - *C* is on the perpendicular bisector of *BE*

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

galois

Theorem: **90=100**

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



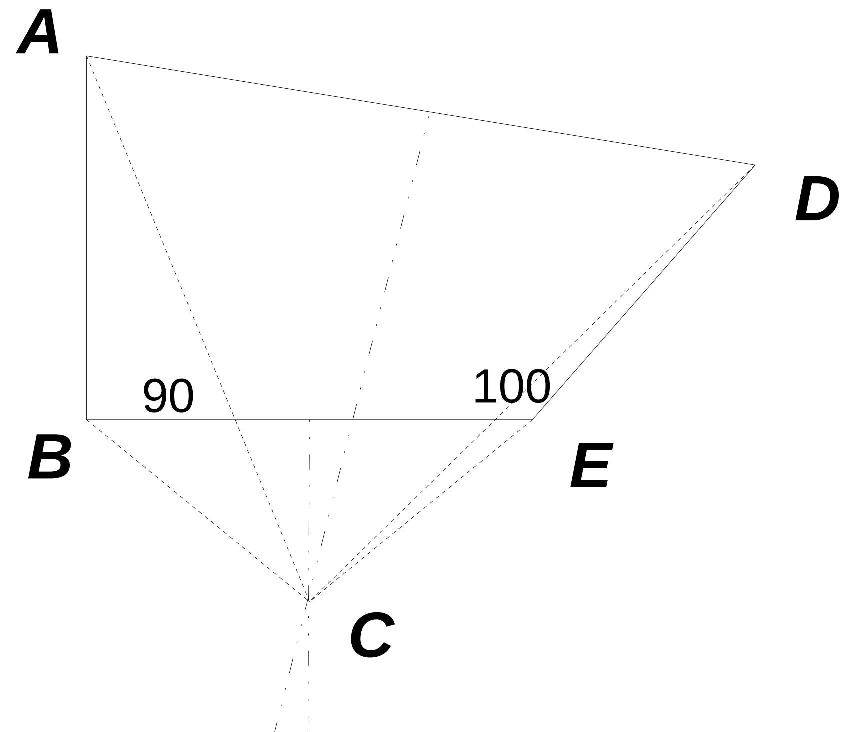
Draw perpendicular bisectors to *BE* and *AD* - they meet at point *C*

$\lvert BC\rvert = \lvert EC\rvert$	- *C* is on the perpendicular bisector of *BE*
$\measuredangle CBE = \measuredangle BEC$	- base angles of isosceles triangle are equal
$\lvert AC\rvert = \lvert DC\rvert$	- $\triangle ACD$ is an isosceles triangle

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$

Draw perpendicular bisectors to BE and AD - they meet at point C



$|BC| = |EC|$ - C is on the perpendicular bisector of BE

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

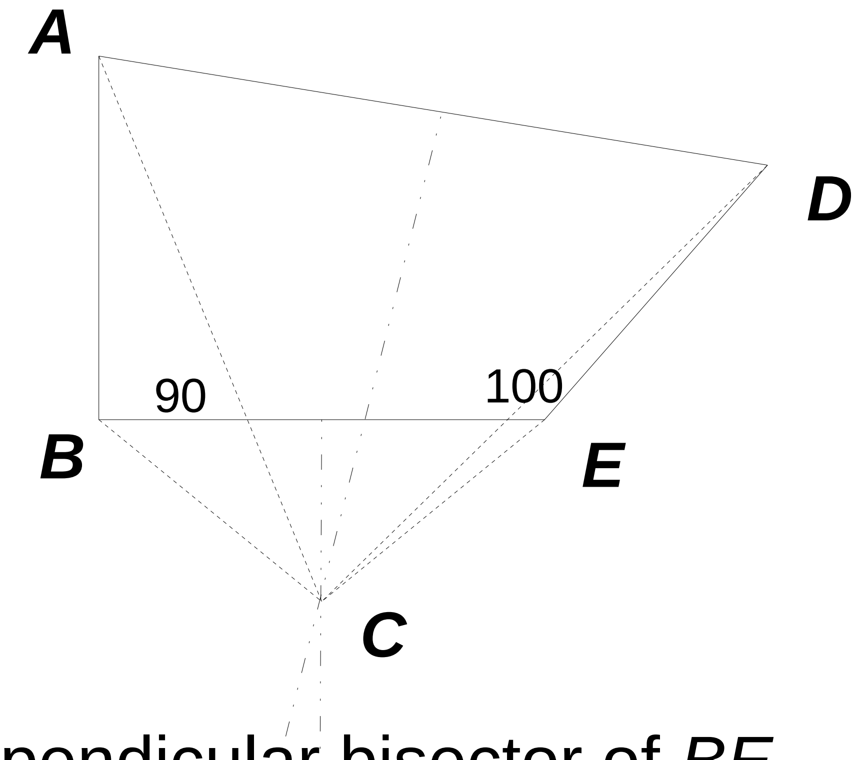
$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

galois

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Draw perpendicular bisectors to *BE* and *AD* - they meet at point *C*

$|BC| = |EC|$ - *C* is on the perpendicular bisector of *BE*

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

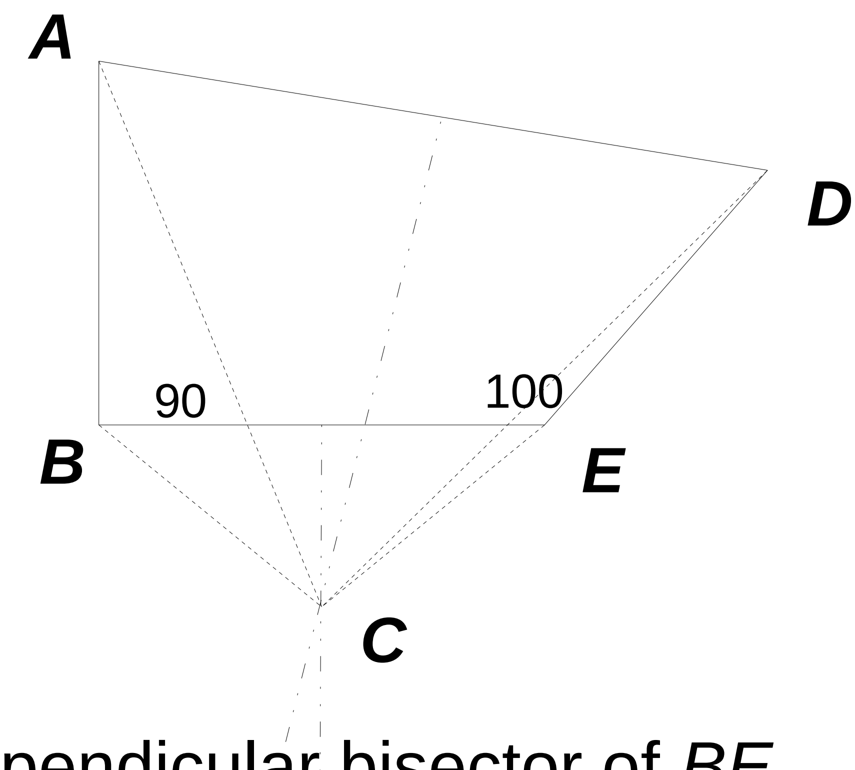
$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Draw perpendicular bisectors to BE and AD - they meet at point C

$|BC| = |EC|$ - C is on the perpendicular bisector of BE

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

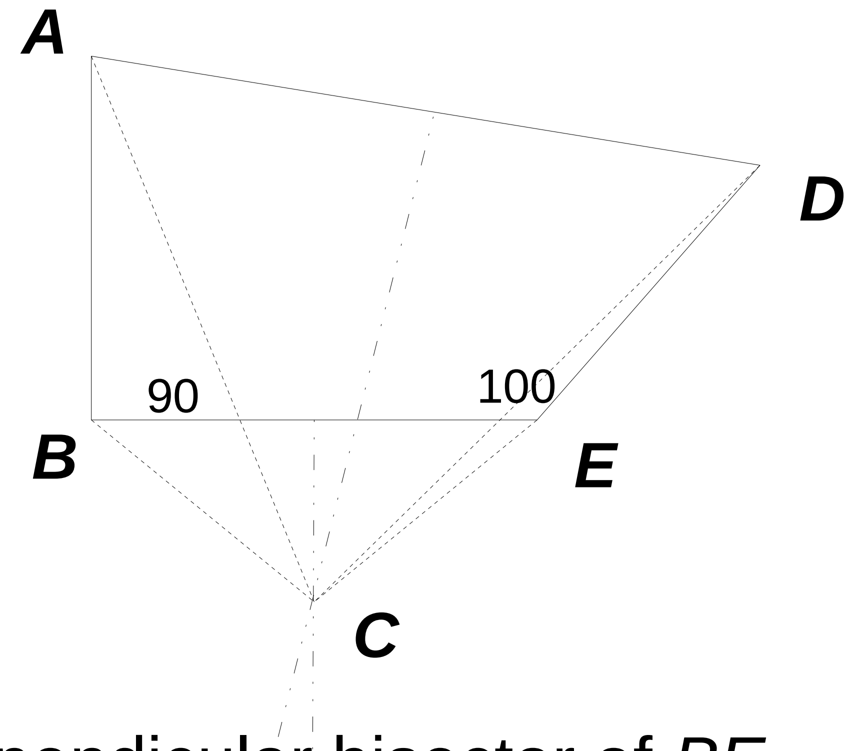
$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

$\measuredangle ABC = \measuredangle ABE + \measuredangle CBE$ and $\measuredangle DEC = \measuredangle DEB + \measuredangle BEC$

Theorem: **90=100**

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Draw perpendicular bisectors to BE and AD - they meet at point C

$|BC| = |EC|$ - C is on the perpendicular bisector of BE

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

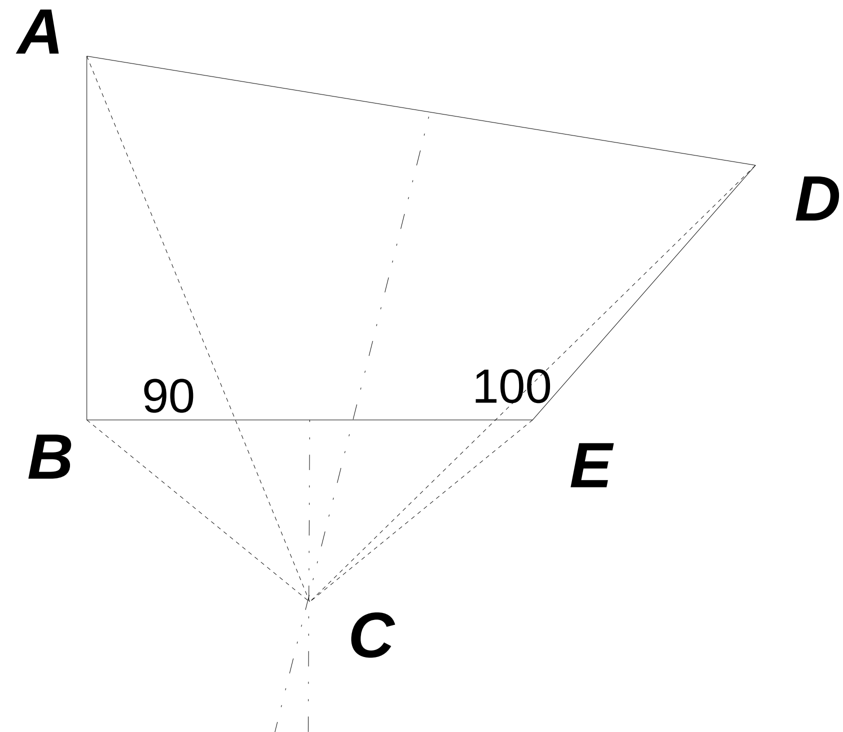
$\measuredangle ABC = \measuredangle ABE + \measuredangle CBE$ and $\measuredangle DEC = \measuredangle DEB + \measuredangle BEC$

$\measuredangle DEC = \measuredangle DEB + \measuredangle CBE$

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$

Draw perpendicular bisectors to BE and AD - they meet at point C



$|BC| = |EC|$ - C is on the perpendicular bisector of BE

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

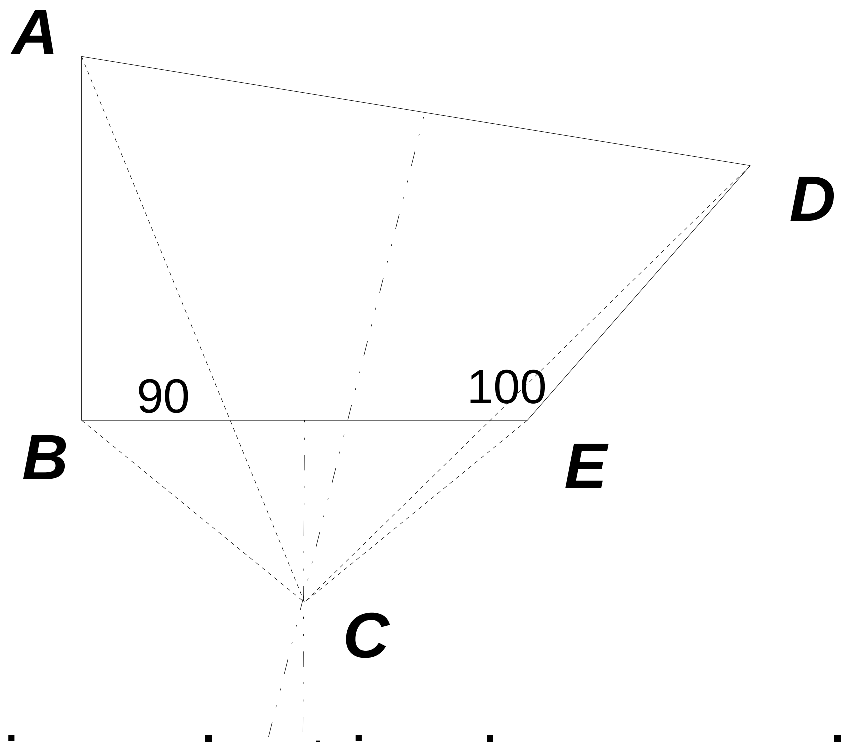
$\measuredangle ABC = \measuredangle ABE + \measuredangle CBE$ and $\measuredangle DEC = \measuredangle DEB + \measuredangle BEC$

$\measuredangle DEC = \measuredangle DEB + \measuredangle CBE$ and $\measuredangle ABC = \measuredangle DEB + \measuredangle CBE$

galois

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Draw perpendicular bisectors to BE and AD - they meet at point C

$\measuredangle CBE = \measuredangle BEC$ - base angles of isosceles triangle are equal

$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

$\measuredangle ABC = \measuredangle ABE + \measuredangle CBE$ and $\measuredangle DEC = \measuredangle DEB + \measuredangle BEC$

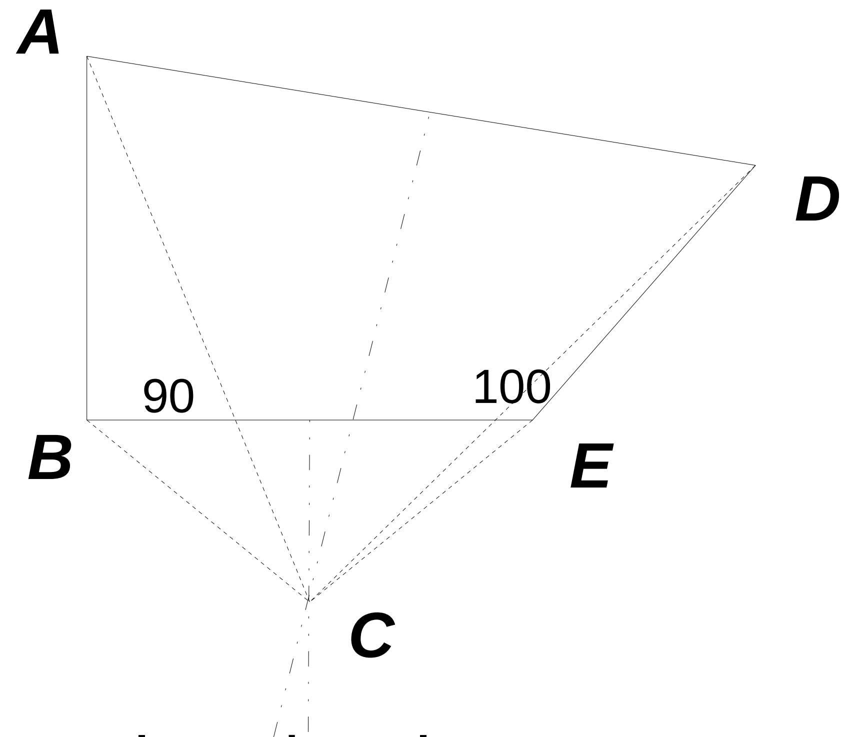
$\measuredangle DEC = \measuredangle DEB + \measuredangle CBE$ and $\measuredangle ABC = \measuredangle DEB + \measuredangle CBE$

$\measuredangle ABE + \measuredangle CBE = \measuredangle DEB + \measuredangle CBE$ - from last two equations

galois

Theorem: 90=100

Proof: Consider 4 sided figure where $\measuredangle ABE = 90^o$ $\measuredangle DEB = 100^o$ $|AB| = |ED|$



Draw perpendicular bisectors to BE and AD - they meet at point C

$|AC| = |DC|$ - $\triangle ACD$ is an isosceles triangle

$\triangle ABC \cong \triangle DEC$ - side-side-side

$\measuredangle ABC = \measuredangle DEC$ - the two triangles are congruent

$\measuredangle ABC = \measuredangle ABE + \measuredangle CBE$ and $\measuredangle DEC = \measuredangle DEB + \measuredangle BEC$

$\measuredangle DEC = \measuredangle DEB + \measuredangle CBE$ and $\measuredangle ABC = \measuredangle DEB + \measuredangle CBE$

$\measuredangle ABE + \measuredangle CBE = \measuredangle DEB + \measuredangle CBE$ - from last two equations

$\measuredangle ABE = \measuredangle DEB$ – from last equation – so **90 = 100**

Do People Actually Use Math Logic?

Yes -

Amazon - verify their implementation of TLS/SSL in AWS

`https://www.youtube.com/watch?v=U40bWY6oVtU#t=33m33s`

Formally verify flight critical software

`https://www.powershow.com/view/3c1c15-YTNiN/Formal_Verification_of_Flight_Critical_Software_powerpoint_ppt_presentation`

Rockwell-Collins - verify components for layered Assurance

https://rockwellcollinsthoughtleadership.wordpress.com/2017/08/22/creating-formally-verified-components-for-layered-assurance-with-an-llvm-to-acl2-translator/

NSA – lots of things – but look at this

`https://www.nsa.gov/resources/everyone/digital-media-center/publications/the-next-wave/assets/files/TNW-19-1.pdf`

Plus lots more

`https://arxiv.org/pdf/1508.07066.pdf`

galois

Set Theory

Set: a collection of definite, distinguishable objects of perception or thought conceived as a whole – Cantor

Nowadays it is known to be possible, logically speaking, to derive practically the whole of known mathematics from a single source, The Theory of Sets – Bourbaki (1930's)

$x \in A$ - object x is a member of set A
$x \notin A$ - object x is not a member of set A
$\emptyset$ - the empty set
$A \subseteq B$ - A is a subset of B
$A \cup B$ - all elements of A and B (union)
$A \cap B$ - all elements common to both A and B (intersection)

See also https://www.math.fsu.edu/~pkirby/mad2104/SlideShow/s1_1.pdf

Set Theory

Relation: a mapping from an ordered pair of set elements to {*true*,*false*} or is a subset of ordered pairs. Thus if (a,b) maps to *true* then $(a,b) \in R$ (R names the subset)
If (a,b) maps to *false* then $(a,b) \notin R$

See also https://www.math.fsu.edu/~pkirby/mad2104/SlideShow/s7_1.pdf

Set Theory

Relation: a mapping from an ordered pair of set elements to {*true*,*false*} or is a subset of ordered pairs. Thus if (a,b) maps to *true* then (a,b) $\in R$ (R names the subset)
If (a,b) maps to *false* then (a,b) $\notin R$

reflexive: a relation R on element $a \in A$ is reflexive iff $(a,a) \in R$

example: consider the relation '*permutation*' (abbrv P) Let x,y be ordered lists. If $(x,y) \in P$ then all elements of x are in y and all elements of y are in x but the order of occurrence may be different in each. Clearly $(x,x) \in P$

Set Theory

Relation: a mapping from an ordered pair of set elements to {*true*,*false*} or is a subset of ordered pairs. Thus if (a,b) maps to *true* then $(a,b) \in R$ (R names the subset)
If (a,b) maps to *false* then $(a,b) \notin R$

reflexive: a relation R on element $a \in A$ is reflexive iff $(a,a) \in R$

example: consider the relation '*permutation*' (abbrv P) Let x,y be ordered lists. If $(x,y) \in P$ then all elements of x are in y and all elements of y are in x but the order of occurrence may be different in each. Clearly $(x,x) \in P$

symmetric: relation R is symmetric iff $(a,b) \in R$ implies $(b,a) \in R$

example: P is symmetric

Set Theory

Relation: a mapping from an ordered pair of set elements to {*true*,*false*} or is a subset of ordered pairs. Thus if (a,b) maps to *true* then $(a,b) \in R$ (R names the subset)
If (a,b) maps to *false* then $(a,b) \notin R$

reflexive: a relation R on element $a \in A$ is reflexive iff $(a,a) \in R$

example: consider the relation '*permutation*' (abbrv P) Let x,y be ordered lists. If $(x,y) \in P$ then all elements of x are in y and all elements of y are in x but the order of occurrence may be different in each. Clearly $(x,x) \in P$

symmetric: relation R is symmetric iff $(a,b) \in R$ implies $(b,a) \in R$

example: P is symmetric

transitive: if $(a,b) \in R$ and $(b,c) \in R$ implies $(a,c) \in R$ then R is transitive

example: P is transitive

Set Theory

Equivalence Relation: a relation that is reflexive, symmetric, and transitive is an *equivalence relation*. All elements involved in an equivalence relation are called an *equivalence class*.

In an equivalence class all objects are equal with respect to the underlying relation.

Thus [1,2,5,3] and [2,5,1,3] are equal w.r.t. permutation

See also https://www.karlin.mff.cuni.cz/~krajicek/mendelson.pdf

Set Theory

Equivalence Relation: a relation that is reflexive, symmetric, and transitive is an *equivalence relation*. All elements involved in an equivalence relation are called an *equivalence class*.

In an equivalence class all objects are equal with respect to the underlying relation.

Thus [1,2,5,3] and [2,5,1,3] are equal w.r.t. permutation

So What?

Suppose you write a program for sorting elements of a list. How do you prove that the program is correct?

Answer:

1. prove the output is a permutation of the input
   The specification of permutation must be shown to produce an equivalence class
2. prove the output elements are in non-decreasing order

Set Theory

Ordering Relation: a relation that is reflexive, anti-symmetric, and transitive.

anti-symmetric: if $(a,b) \in R$ then $(b,a) \notin R$

example: $a,b \in \mathbb{N}^{+}$, R is $\leq$

Note: < is not reflexive, but $\leq$ defines a partial order

Set Theory

Ordering Relation: a relation that is reflexive, anti-symmetric, and transitive.

anti-symmetric: if $(a,b) \in R$ then $(b,a) \notin R$

example: $a,b \in \mathbb{N}^+$, R is $\leq$

Note: $<$ is not reflexive, but $\leq$ defines a partial order

So What?

Suppose you write a program for sorting elements of a list. How do you prove that the program is correct?

Answer:

1. prove the output is a permutation of the input
2. prove the output elements are in non-decreasing order
   Specification of non-decreasing must be shown to produce a partial order

Set Theory

Function: a relation f where if $(a,b) \in f$ and $(a,c) \in f$ then $b = c$

See also https://www.karlin.mff.cuni.cz/~krajicek/mendelson.pdf

See also https://www.math.fsu.edu/~pkirby/mad2104/SlideShow/s1_2.pdf

Set Theory

Function: a relation f where if $(a,b) \in f$ and $(a,c) \in f$ then $b = c$

Formula: a function that maps to *true* or *false*, depending on input, defined recursively as follows:

1. for any variables x and y, $x \in y$ and $x = y$ are formulas
2. If S and T are formulas and x is any variable, then each of the following is a formula (add parens for clarity):

$S \rightarrow T$	- S implies T
$S \leftrightarrow T$	- S iff T – could have value *false*
$S \equiv T$	- S equivalent to T – is always *true*
$S \wedge T$	- *true* if S and T are *true*
$S \vee T$	- *true* if S or T is *true*
$\neg S$	- *true* if S is *false*
$\forall x, S$	- S is *true* for all values of x
$\exists x, T$	- there is an x for which T *is true*

example: $f(x,y) = \forall y, (x \in y)$ – since x is free this is a *condition* on x and denoted $S(x)$

example: $10 \neq 20$ but $10 \equiv 20 \bmod 5$

Now What?

We would like to develop a system based in set theory that enables proofs of properties about objects in sets

However, we would like to avoid systems that are unsound - that is, derives a falsehood

Now What?

We would like to develop a system based in set theory that enables proofs of properties about objects in sets

However, we would like to avoid systems that are unsound - that is, derives a falsehood

example: there is a town with one barber who shaves all those, and only those, who do not shave themselves. Who shaves the barber?

Now What?

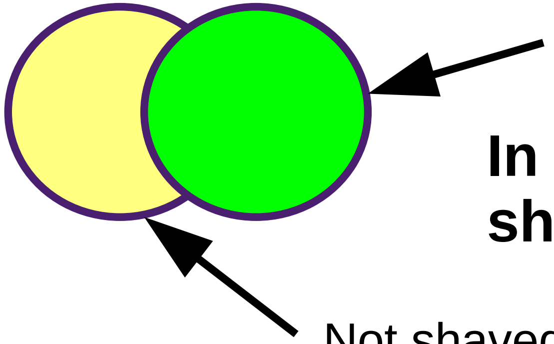
We would like to develop a system based in set theory that enables proofs of properties about objects in sets

However, we would like to avoid systems that are unsound - that is, derives a falsehood

example: there is a town with one barber who shaves all those, and only those, who do not shave themselves. Who shaves the barber?

The barber can't shave himself because he shaves only people who do not shave themselves.

The barber can't have someone else shave him because then he must shave himself.

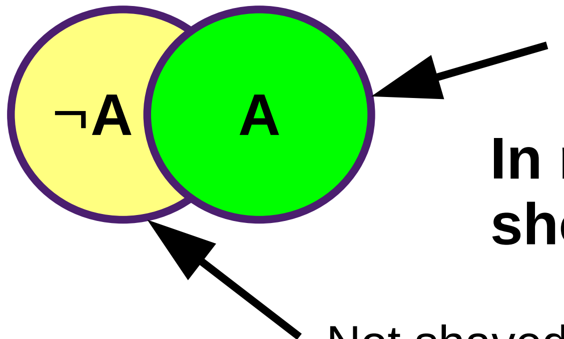


Shaved by the barber – barber can't be in this group

In naïve set theory it looks like there should be a solution

Not shaved by the barber – shave themselves

Now What?



Shaved by the barber – barber can't be in this group

In naïve set theory it looks like there should be a solution

Not shaved by the barber – shave themselves

Let x be the barber – in which set is the barber?
If $x \in A$ then $x \in \neg A$ (barber not allowed to shave self)
If $x \in \neg A$ then $x \in A$ (barber must shave self)

barber shaves exactly everyone
who does not shave himself

$\exists x$ (person(x) $\wedge$ $\forall y$ ((person(y) $\rightarrow$ (shaves(x,y) $\leftrightarrow$ $\neg$shaves(y,y))))
But set of y includes the barber who is x. Then we have
shaves(x,x) $\leftrightarrow$ $\neg$shaves(x,x) which is a contradiction

Now What?

We would like to develop a system based in set theory that enables proofs of properties about objects in sets.

To avoid inconsistencies allowed sets are built by extending known sets. The base set that is known is the empty set $\varnothing$

Axioms:

1. A and B are sets: $\forall x\, ((x \in A \leftrightarrow x \in B) \rightarrow A{=}B)$
2. A is a set and $P(x)$ is a formula: there is a set B such that $\forall x\, ((x \in B \leftrightarrow x \in A)$ and $P(x)$ is true). In symbols:
   $$B = \{\, x \in A \mid P(x) \,\} \qquad (1)$$
   Note: any set where x is not restricted to a set A – i.e. $B = \{\, x \mid P(x) \,\}$ - is not allowed. Then B could be a set of all x such that $x \neq x$. But (1) says B is not a member of A.
3. There exists a set $\varnothing$ such that $\forall x$, x a set, $x \notin \varnothing$
   (note: empty set is start of constructing a set)

galois

Axioms:

4. If A and B are sets there exists another set denoted $\{A,B\}$ that contains all the members of A and B and no others
5. $\mathscr{L}$ is a collection of sets. There exists set B such that $x \in B$ iff $\exists A \in \mathscr{L}$ such that $x \in A$ (set union)
6. If A is a set there exists a set B (power set) such that $x \in B$ iff $x \subseteq A$
7. Let A be a set. Let $f(x, y)$ be a formula which associates to each element x of A an element y in such a way that whenever both $f(x, y)$ and $f(x, z)$ hold true, $y = z$. Then there exists a set B which contains all elements y such that $f(x, y)$ holds true for some $x \in A$.
8. Every non-empty set A contains an element B such that $A \cap B = \varnothing$ (no common elements)
   $x \neq \varnothing \rightarrow \exists y\,(\, y \in x$ and $y \cap x = \varnothing\,)$
9. For every set $\mathscr{L}$ of non-empty sets there is a function f which associates to every set A in $\mathscr{L}$ an element $a \in A$

Model Theory:

Meaning of an expression (semantics) is determined from mechanical systems (proof systems) which are syntactic elements.

We are concerned with propositional logic and 1st order logic.

Propositional Logic

A proposition is true or false or undetermined

Some insects can fly
The sun sets after 3PM on every summer day
The Reds will win the World Series this year

Inferences

P implies *Q* is a proposition which is true if
P is false or *P* is true and *Q* is true

Truth Tables

P	Q	f
F	F	T
F	T	T
T	F	F
T	T	T

$f = P \rightarrow Q$

P	Q	f
F	F	F
F	T	T
T	F	T
T	T	T

$\mathrm{f} = P \vee Q$

P	Q	f
F	F	F
F	T	F
T	F	F
T	T	T

$f = P \wedge Q$

P	Q	f
F	F	F
F	T	T
T	F	T
T	T	F

$f = P \oplus Q$

P	Q	f
F	F	T
F	T	F
T	F	F
T	T	T

$f = P \leftrightarrow Q$

$$P \rightarrow Q \equiv \neg P \vee Q$$

$$(P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P)$$

Model Theory:

$$(P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P) \textbf{ ?}$$

Model Theory:

$(P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P)$ **?** /* call this X */

P	Q	$P \rightarrow Q$	$\neg Q \rightarrow \neg P$	X
false	*false*	*true*	*true*	*true*
false	*true*	*true*	*true*	*true*
true	*false*	*false*	*false*	*true*
true	*true*	*true*	*true*	*true*

Model Theory:

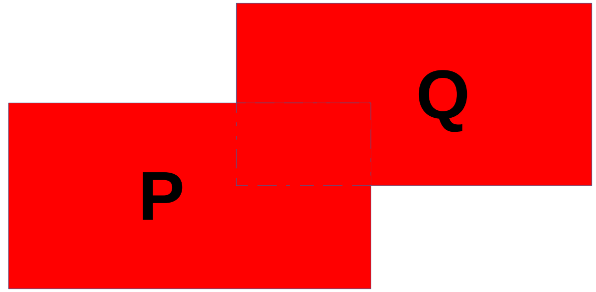
$\models (P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P)$ **?** /* call this X */

P	***Q***	***P→Q***	***¬Q→¬P***	**X**
false	*false*	*true*	*true*	*true*
false	*true*	*true*	*true*	*true*
true	*false*	*false*	*false*	*true*
true	*true*	*true*	*true*	*true*

This means tautology in this context

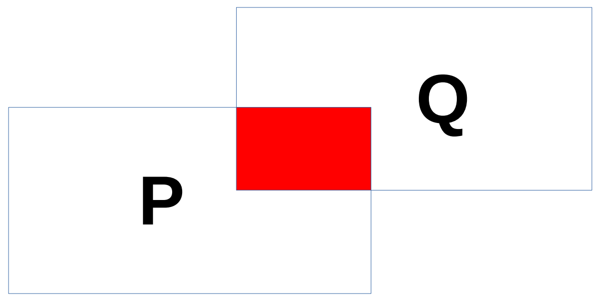
Logic and Sets

$P \lor Q$



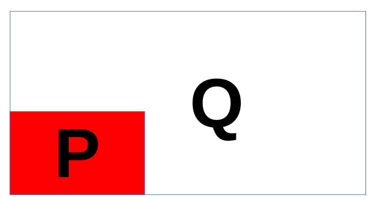
$P \cup Q$

$P \land Q$



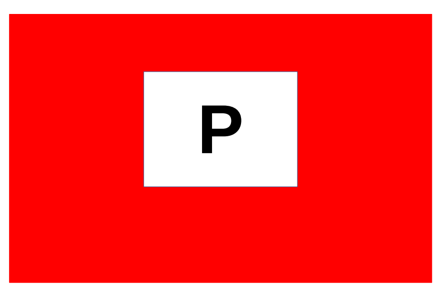
$P \cap Q$

$P \rightarrow Q$



$P \subseteq Q$

$\neg P$



$\overline{P}$

galois

Proof System:

Set of axioms (statements considered true)

Set of inference rules

New statements are deduced by constructing a sequence of steps from axioms, deduced facts using inference rules

In use: user adds premises to the axioms
the user believes these to be true

Soundness: statements obtained from the application of inference rules in a sequence of steps are true provided all axioms and premises are true

Completeness: all true statements are derivable

See also https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwj95d-cm8SCAxU0hIkEHdQmCr4QFnoECBoQAQ&url=https%3A%2F%2Fwww3.cs.stonybrook.edu%2F~cse541%2Fchapter7.pdf&usg=AOvVaw3MaVXRO7jL4C-xi_OtKFiS&opi=89978449

galois

Example:

Proposition types (quantifiers):

A: *Every S is P* is true in world w iff $S = S \cap P$ ($S \subseteq P$)
Everything in w signified by *S* is something signified in w by *S* and *P*

I: *Some S is P* is true in world w iff ($S \cap P \neq \emptyset$)
There is a *T* such that all signified in w by *S* and *P* is something that is signified in w by *P* and *T*

E: *No S is P* is true in world w iff $\neg(S \cap P \neq \emptyset)$
Some *S* is *P* is false in w

O: *Some S is not P* is true in world w iff $\neg(S = S \cap P)$
Every *S* is *P* is false in w

Syllogism: $(p \wedge q) \rightarrow r$ where *p*, *q*, *r* are of type A,E,I, or O

For all *p*, either *p* or $\neg p$ is always true

See also https://thedecisionlab.com/reference-guide/philosophy/syllogism

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow$ r infer $(\neg r \wedge q) \rightarrow \neg p$

R2: From $(p \wedge q) \rightarrow$ r infer $(q \wedge p) \rightarrow r$

R3: From no Q is P infer no P is Q

R4: From $(p \wedge q) \rightarrow$ no Q is P infer $(p \wedge q) \rightarrow$ some Q is not P

Axioms:

A1: (every Q is P $\wedge$ every P is S) $\rightarrow$ every Q is S

A2: (every Q is P $\wedge$ no P is S) $\rightarrow$ no Q is S

Syllogism: $(p \wedge q) \rightarrow r$ where p, q ,r are of type A,E,I, or O

Prove: (every P is M $\wedge$ no S is M) $\rightarrow$ some S is not P

galois

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow r$ infer $(\neg r \wedge q) \rightarrow \neg p$

R2: From $(p \wedge q) \rightarrow r$ infer $(q \wedge p) \rightarrow r$

R3: From no *Q* is *P* infer no *P* is *Q*

R4: From $(p \wedge q) \rightarrow$ no *Q* is *P* infer $(p \wedge q) \rightarrow$ some *Q* is not *P*

Axioms:

A1: (every *Q* is *P* $\wedge$ every *P* is *S*) $\rightarrow$ every *Q* is *S*

A2: (every *Q* is *P* $\wedge$ no *P* is *S*) $\rightarrow$ no *Q* is *S*

Syllogism: $(p \wedge q) \rightarrow r$ where *p*, *q* ,*r* are of type A,E,I, or O

Prove: (every *P* is *M* $\wedge$ no *S* is *M*) $\rightarrow$ some *S* is not *P*

1. (every *P* is *M* $\wedge$ no *M* is *S*) $\rightarrow$ no *P* is *S* by axiom A2

Example:

Inference Rules:

R1: From $(p \wedge q) \to$ r infer $(\neg r \wedge q) \to \neg p$

R2: From $(p \wedge q) \to$ r infer $(q \wedge p) \to r$

R3: From no Q is P infer no P is Q

R4: From $(p \wedge q) \to$ no Q is P infer $(p \wedge q) \to$ some Q is not P

Axioms:

A1: (every Q is P $\wedge$ every P is S) $\to$ every Q is S

A2: (every Q is P $\wedge$ no P is S) $\to$ no Q is S

Syllogism: $(p \wedge q) \to r$ where p, q ,r are of type A,E,I, or O

Prove: (every P is M $\wedge$ no S is M) $\to$ some S is not P

1. (every P is M $\wedge$ no M is S) $\to$ no P is S by axiom A2
2. (every P is M $\wedge$ no S is M) $\to$ no S is P by rule R3

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow r$ infer $(\neg r \wedge q) \rightarrow \neg p$

R2: From $(p \wedge q) \rightarrow r$ infer $(q \wedge p) \rightarrow r$

R3: From no *Q* is *P* infer no *P* is *Q*

R4: From $(p \wedge q) \rightarrow$ no *Q* is *P* infer $(p \wedge q) \rightarrow$ some *Q* is not *P*

Axioms:

A1: (every *Q* is *P* $\wedge$ every *P* is *S*) $\rightarrow$ every *Q* is *S*

A2: (every *Q* is *P* $\wedge$ no *P* is *S*) $\rightarrow$ no *Q* is *S*

Syllogism: $(p \wedge q) \rightarrow r$ where *p*, *q* ,*r* are of type A,E,I, or O

Prove: (every *P* is *M* $\wedge$ no *S* is *M*) $\rightarrow$ some *S* is not *P*

1. (every *P* is *M* $\wedge$ no *M* is *S*) $\rightarrow$ no *P* is *S* by axiom A2
2. (every *P* is *M* $\wedge$ no *S* is *M*) $\rightarrow$ no *S* is *P* by rule R3
3. (every *P* is *M* $\wedge$ no *S* is *M*) $\rightarrow$ some *S* is not *P* rule R4

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow$ r infer $(\neg r \wedge q) \rightarrow \neg p$

R2: From $(p \wedge q) \rightarrow$ r infer $(q \wedge p) \rightarrow r$

R3: From no Q is P infer no P is Q

R4: From $(p \wedge q) \rightarrow$ no Q is P infer $(p \wedge q) \rightarrow$ some Q is not P

Axioms:

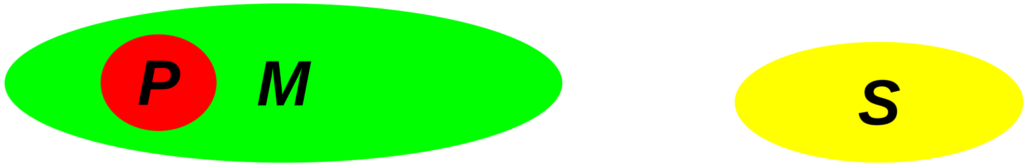
A1: (every Q is P $\wedge$ every P is S) $\rightarrow$ every Q is S

A2: (every Q is P $\wedge$ no P is S) $\rightarrow$ no Q is S

Syllogism: $(p \wedge q) \rightarrow r$ where p, q ,r are of type A,E,I, or O

Prove: (every P is M $\wedge$ no S is M) $\rightarrow$ some S is not P

1. (every P is M $\wedge$ no M is S) $\rightarrow$ no P is S by axiom A2
2. (every P is M $\wedge$ no S is M) $\rightarrow$ no S is P by rule R3
3. (every P is M $\wedge$ no S is M) $\rightarrow$ some S is not P rule R4



galois

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow$ r infer $(\neg r \wedge q) \rightarrow \neg p$

R2: From $(p \wedge q) \rightarrow$ r infer $(q \wedge p) \rightarrow r$

R3: From no Q is P infer no P is Q

R4: From $(p \wedge q) \rightarrow$ no Q is P infer $(p \wedge q) \rightarrow$ some Q is not P

Axioms:

A1: (every Q is P $\wedge$ every P is S) $\rightarrow$ every Q is S

A2: (every Q is P $\wedge$ no P is S) $\rightarrow$ no Q is S

Syllogism: $(p \wedge q) \rightarrow r$ where p, q ,r are of type A,E,I, or O

Prove: (some M is A $\wedge$ some C is A) $\rightarrow$ every M is C is false!

Example:

Inference Rules:

R1:	From $(p \wedge q) \rightarrow$ r	infer	$(\neg r \wedge q) \rightarrow \neg p$
R2:	From $(p \wedge q) \rightarrow$ r	infer	$(q \wedge p) \rightarrow r$
R3:	From no Q is P	infer	no P is Q
R4:	From $(p \wedge q) \rightarrow$ no Q is P	infer	$(p \wedge q) \rightarrow$ some Q is not P

Axioms:

A1: (every Q is P $\wedge$ every P is S) $\rightarrow$ every Q is S

A2: (every Q is P $\wedge$ no P is S) $\rightarrow$ no Q is S

Syllogism: $(p \wedge q) \rightarrow r$ where p, q ,r are of type A,E,I, or O

Prove: (some M is A $\wedge$ some C is A) $\rightarrow$ every M is C is false!

(some M is A $\wedge$ some C is A) $\rightarrow$ some M is not C is true

Example:

Inference Rules:

R1: From $(p \wedge q) \rightarrow$ r infer $(\neg r \wedge q) \rightarrow \neg p$
R2: From $(p \wedge q) \rightarrow$ r infer $(q \wedge p) \rightarrow r$
R3: From no Q is P infer no P is Q
R4: From $(p \wedge q) \rightarrow$ no Q is P infer $(p \wedge q) \rightarrow$ some Q is not P

Axioms:

A1: (every Q is P $\wedge$ every P is S) $\rightarrow$ every Q is S
A2: (every Q is P $\wedge$ no P is S) $\rightarrow$ no Q is S

Syllogism: $(p \wedge q) \rightarrow r$ where p, q ,r are of type A,E,I, or O

Prove: (some M is A $\wedge$ some C is A) $\rightarrow$ every M is C is false!
(some M is A $\wedge$ some C is A) $\rightarrow$ some M is not C is true

Let M be a man, let C be a cow, let A be an animal
the man and the cow are animals but
the man is not a cow

galois

Conjunctive Normal Form:

variables: $a,b,c..$ values are 0,1,unassigned

literals: $a, \neg a$ a has value 1 iff $\neg a$ has value 0

connectives: $\wedge \vee \neg$

clause: $(a \vee \neg b \vee c)$

formula: conjunction of clauses

$(a \vee \neg b \vee c) \wedge (b \vee \neg c \vee \neg d) \wedge (\neg a \vee d) \wedge (\neg c \vee \neg d)$

Does there exist an assignment of values to variables (model, interpretation) such that a given formula has value 1?

If so, the formula is satisfiable

Above is satisfiable – here is a model: a, b, $d = 1$; $c = 0$

See also gauss.ececs.uc.edu/Courses/c6021/references/algs-for-sat.pdf

galois

Set Representation:

$$\varphi = \{\{\neg v_0, v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_0, v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$$

Resolution: must have: no *w* such that *w* is in one and $\neg w$ is in other

$$\varphi_v = \{ c - \{\neg v\} \mid c \in \varphi, v \notin c \}$$
$$\varphi_{\neg v} = \{ c - \{v\} \mid c \in \varphi, \neg v \notin c \}$$
$$\{ c - \{\neg v, v\} \mid c \in \varphi, v, \neg v \notin c \}$$

Axioms:

$\varphi = \{\ldots\ \varnothing\ \ldots\}$ — φ is unsatisfiable

$\varphi = \{\}$ — φ is satisfiable

Set Representation:

$$\varphi = \{\underset{1}{\{\neg v_0,\ v_1,\ \neg v_2\}},\ \underset{2}{\{\neg v_1,\ \neg v_3\}},\ \underset{3}{\{v_0,\ v_3,\ \neg v_4,\ \neg v_5\}},\ \underset{4}{\{v_3\}}\}$$

Resolution: must have: no *w* such that *w* is in one and $\neg w$ is in other

$$\left.\begin{array}{l} \varphi_v = \{\ c - \{\neg v\} \mid c \in \varphi,\ v \notin c\ \} \\ \varphi_{\neg v} = \{\ c - \{v\} \mid c \in \varphi,\ \neg v \notin c\ \} \end{array}\right\} \rightarrow \{\ c - \{\neg v, v\} \mid c \in \varphi,\ v, \neg v \notin c\ \}$$

All Resolutions:

$$\mathbf{V}_0 \Rightarrow \{\underset{1,3}{\{v_1, \neg v_2,\ v_3,\ \neg v_4,\ \neg v_5\}},\ \underset{2}{\{\neg v_1,\ \neg v_3\}},\ \underset{4}{\{v_3\}}\}$$

$$\mathbf{V_3} \Rightarrow \{\underset{2,4}{\{\neg v_1\}}\}$$

$$\mathbf{V}_1 \Rightarrow \{\ \}$$ hence φ is satisfiable

Axioms:

$\varphi = \{\ldots\ \varnothing\ \ldots\}$ φ is unsatisfiable

$\varphi = \{\ \}$ φ is satisfiable

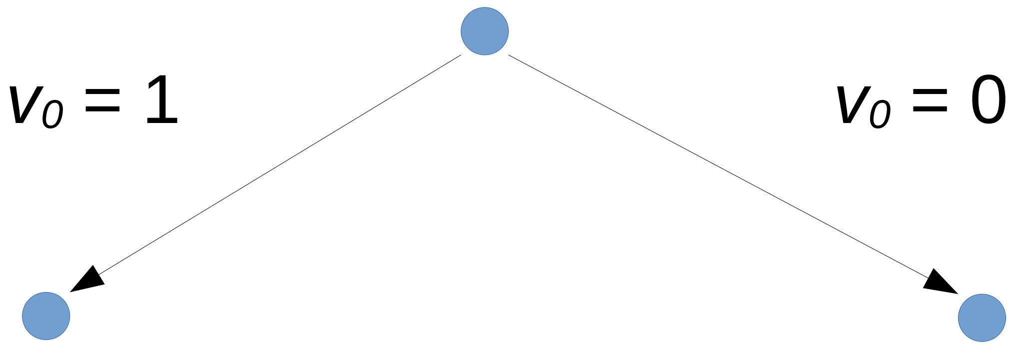
Set Representation:

$$\varphi = \{\{\neg v_0, v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_0, v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$$

Resolution:

$$\varphi_{v} = \{\, c - \{\neg v\} \mid c \in \varphi,\ v \notin c \,\}$$
$$\varphi_{\neg v} = \{\, c - \{v\} \mid c \in \varphi,\ \neg v \notin c \,\}$$

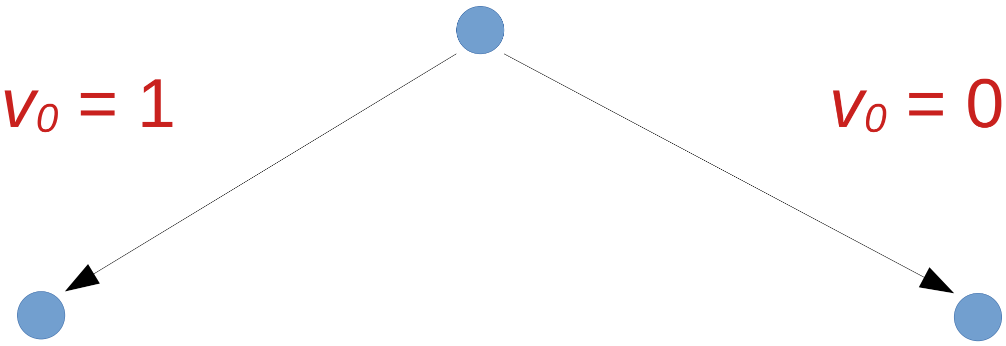
$$\{\{\neg v_0, v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_0, v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$$



$$\{\{v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_3\}\} \qquad \{\{\neg v_1, \neg v_3\}, \{v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$$

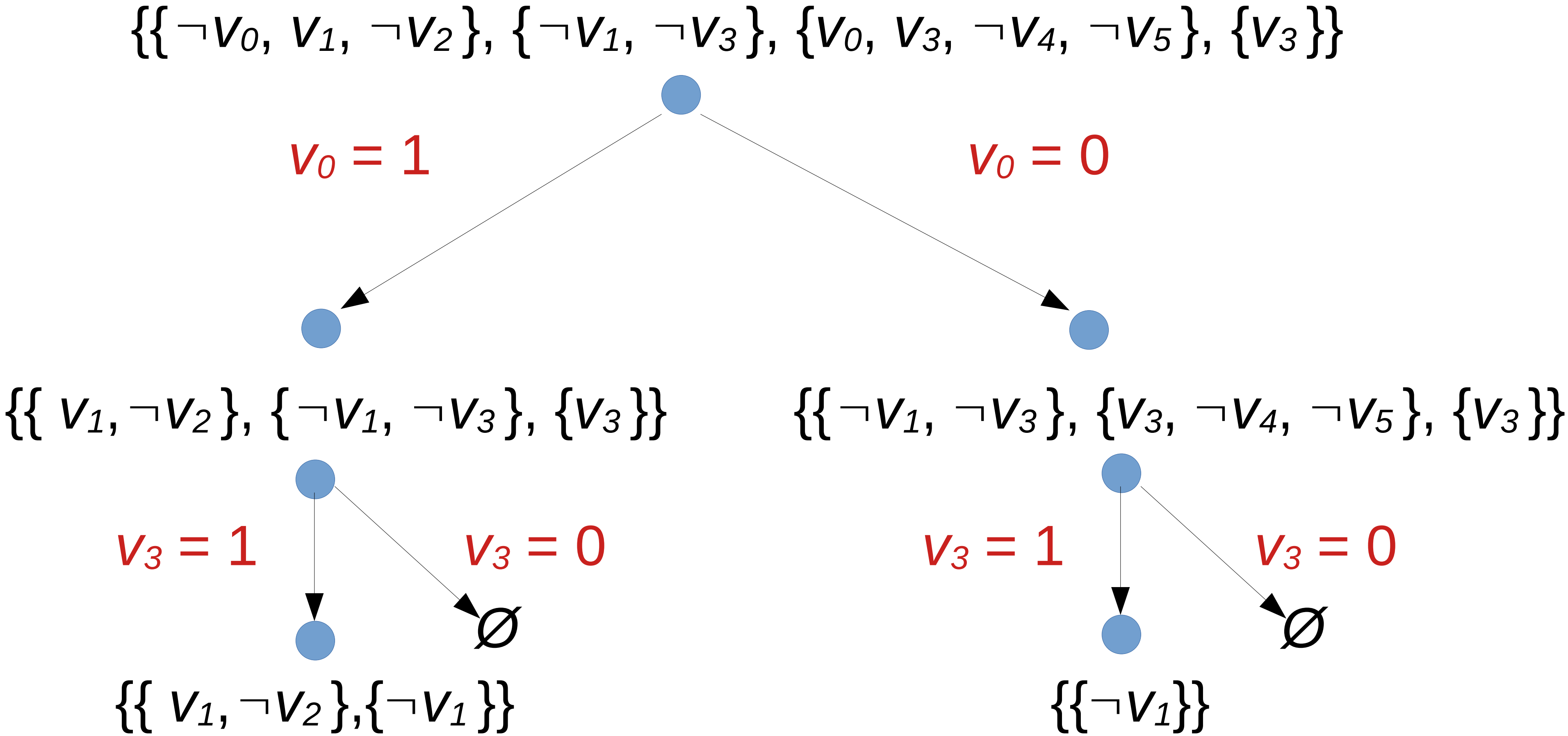
$$\varphi_{v_0} \qquad \varphi_{\neg v_0}$$

$\{\{\neg v_0, v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_0, v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$

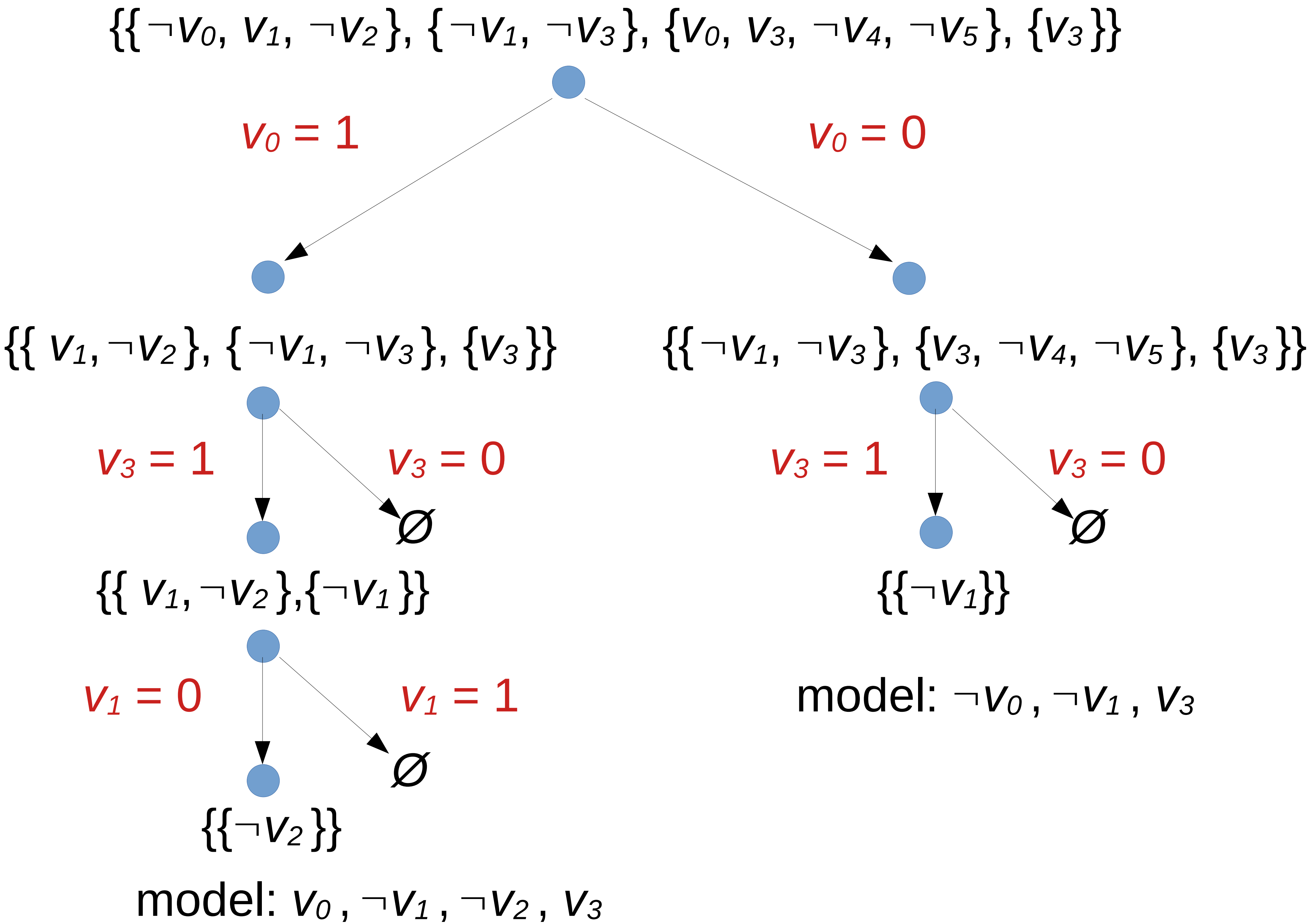


$\{\{v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_3\}\}$

$\{\{\neg v_1, \neg v_3\}, \{v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$



model: $\neg v_0, \neg v_1, v_3$

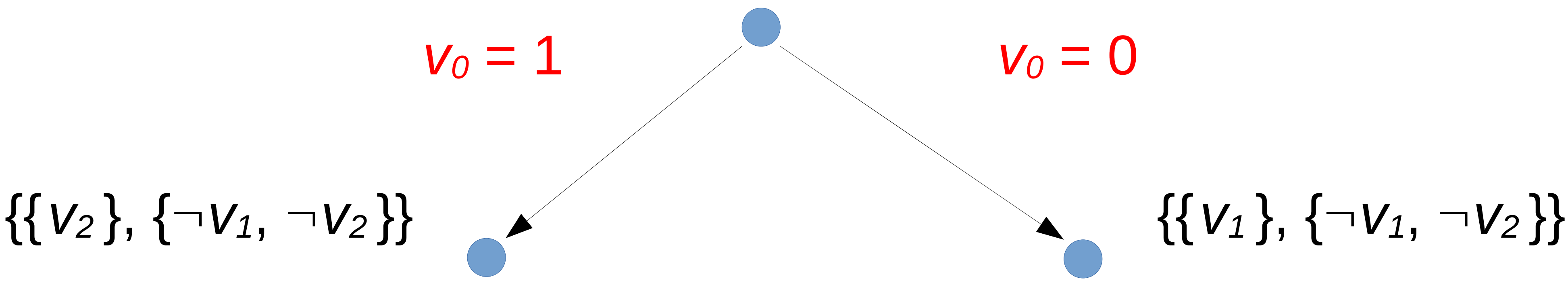

$\{\{\neg v_0, v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_0, v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$
$v_0 = 1$
$v_0 = 0$
$\{\{v_1, \neg v_2\}, \{\neg v_1, \neg v_3\}, \{v_3\}\}$
$\{\{\neg v_1, \neg v_3\}, \{v_3, \neg v_4, \neg v_5\}, \{v_3\}\}$
$v_3 = 1$
$v_3 = 0$
$\varnothing$
$v_3 = 1$
$v_3 = 0$
$\varnothing$
$\{\{v_1, \neg v_2\}, \{\neg v_1\}\}$
$\{\{\neg v_1\}\}$
$v_1 = 0$
$v_1 = 1$
$\varnothing$
model: $\neg v_0, \neg v_1, v_3$
$\{\{\neg v_2\}\}$
model: $v_0, \neg v_1, \neg v_2, v_3$

Example:

$$\varphi = \{\{v_0, v_1\}, \{\neg v_0, v_2\}, \{\neg v_1, \neg v_2\}\}$$

Example:

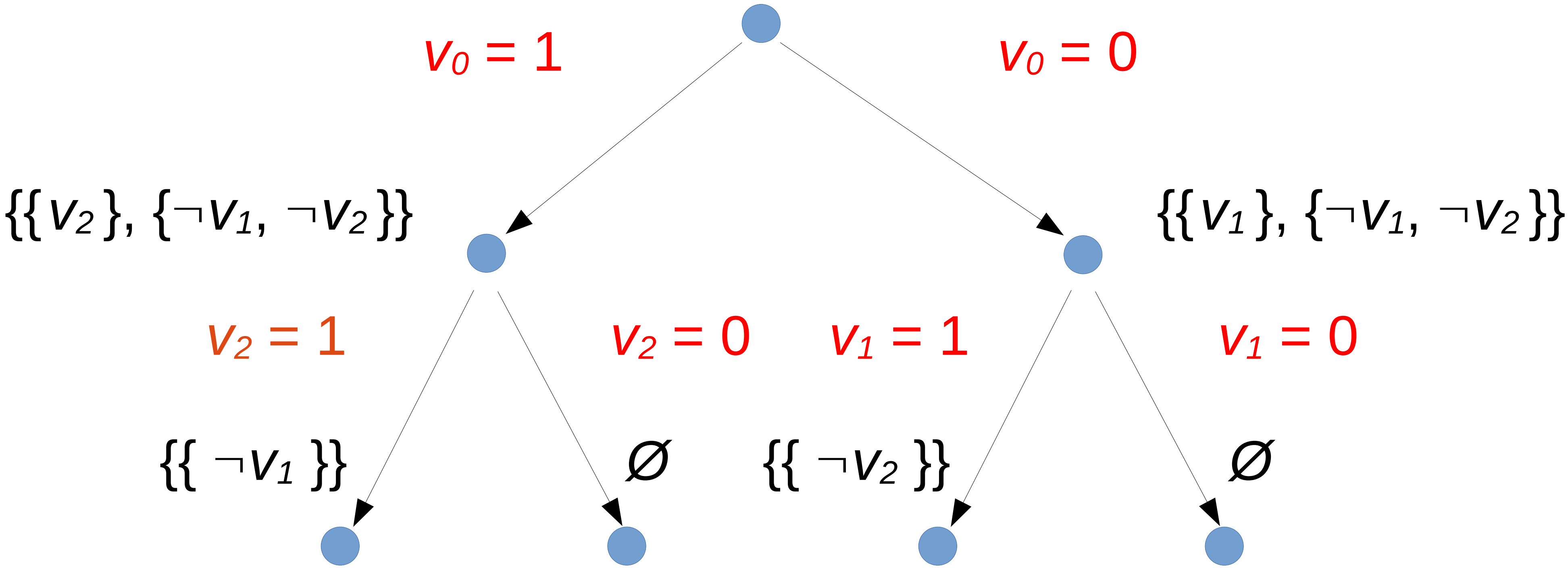
$$\varphi = \{\{v_0, v_1\}, \{\neg v_0, v_2\}, \{\neg v_1, \neg v_2\}\}$$



galois

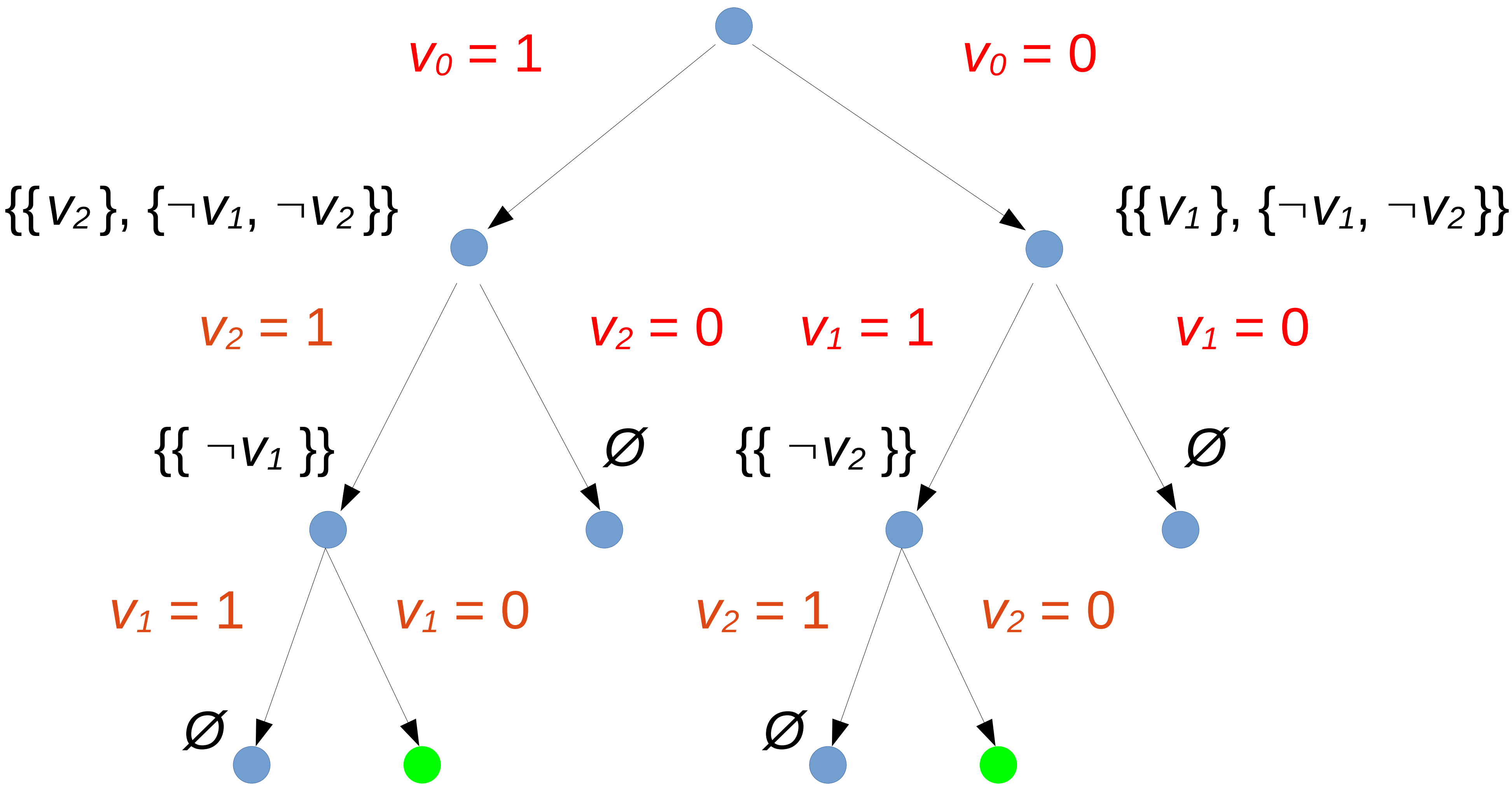
Example:

$$\varphi = \{\{v_0, v_1\}, \{\neg v_0, v_2\}, \{\neg v_1, \neg v_2\}\}$$



Example:

$$\varphi = \{\{v_0, v_1\}, \{\neg v_0, v_2\}, \{\neg v_1, \neg v_2\}\}$$



Solutions: $\{\{v_0, \neg v_1, v_2\}, \{\neg v_0, v_1, \neg v_2\}\}$ as DNF (sum of products)

Example: door 1 door 2 door 3

door 1: The way out of the cave is behind this door

door 2: If you pass through this door you are doomed to remain in the cave forever

door 3: If you pass Through the Green door you are doomed to remain in the cave forever

An explorer scout troop that is exploring a cave has become lost. While looking for a way out they stumble upon three doors with signs pasted on their faces. A note found nearby says that each sign is *true* or *false*, at least one sign is *true* and at least one sign is *false*, doors do not open from the other side, and there is a way out behind at least one door. Can the troop determine through which door they should pass through to get out of the cave?

Setup: door 1 door 2 door 3

door 1	door 2	door 3
The way out of the cave is behind this door	If you pass through this door you are doomed to remain in the cave forever	If you pass Through the Green door you are doomed to remain in the cave forever

Choose variables:

a: true iff there is a path to freedom behind door 1
b: true iff there is a path to freedom behind door 2
c: true iff there is a path to freedom behind door 3
d: true iff sign on door 1 is true
e: true iff sign on door 2 is true
f: true iff sign on door 3 is true

Setup:

Represent Constraints:

There is a way out behind at least one door

$(a \vee b \vee c)$

galois

Setup:

Represent Constraints:

There is a way out behind at least one door

$(a \vee b \vee c)$

At least one sign is *true* and one is *false*

$(d \wedge \neg e \wedge \neg f) \vee (d \wedge \neg e \wedge f) \vee (d \wedge e \wedge \neg f) \vee$
$(\neg d \wedge e \wedge f) \vee (\neg d \wedge \neg e \wedge f) \vee (\neg d \wedge e \wedge \neg f)$

Setup:

Represent Constraints:

There is a way out behind at least one door

$(a \vee b \vee c)$

At least one sign is *true* and one is *false*

$(d \wedge \neg e \wedge \neg f\,) \vee (d \wedge \neg e \wedge f\,) \vee (d \wedge e \wedge \neg f\,) \vee$
$(\neg d \wedge e \wedge f\,) \vee (\neg d \wedge \neg e \wedge f\,) \vee (\neg d \wedge e \wedge \neg f\,)$

But $d \equiv a$, $e \equiv \neg b$, $\mathrm{f} \equiv \neg b$ so this becomes

$(a \wedge b \wedge b) \vee (a \wedge b \wedge \neg b) \vee (a \wedge \neg b \wedge b) \vee$
$(\neg a \wedge \neg b \wedge \neg b) \vee (\neg a \wedge b \wedge \neg b) \vee (\neg a \wedge \neg b \wedge b)$

$= (a \wedge b) \vee (\neg a \wedge \neg b)$
$= (a \vee \neg b) \wedge (\neg a \vee b)$

Total Formula:

$\varphi = (a \vee \neg b) \wedge (\neg a \vee b) \wedge (a \vee b \vee c)$

For every model of φ there is a way out, at least one door sign is true and at least one is false. How does this help?

Interpretation:

Total Formula:

$$\varphi = (a \lor \neg b) \land (\neg a \lor b) \land (a \lor b \lor c)$$

For every model of φ there is a way out, at least one door sign is true and at least one is false. How does this help?

3 Models: $a = b = c =$ true, $a = b =$ false and $c =$ true,
$a = b =$ true *and* $c =$ false

So there is no definite way out – maybe door 3, maybe not

Interpretation:

Total Formula:

$\varphi = (a \lor \neg b) \land (\neg a \lor b) \land (a \lor b \lor c)$

For every model of φ there is a way out, at least one door sign is true and at least one is false. How does this help?

3 Models: $a = b = c =$ true, $a = b =$ false and $c =$ true,
$a = b =$ true *and* $c =$ false

So there is no definite way out – maybe door 3, maybe not

But what if the problem states only one exit is possible? What clauses to add to the “total formula” to accommodate the change?

$(\neg a \lor \neg b) \land (\neg a \lor \neg c) \land (\neg b \lor \neg c)$

If a is true then b must be false or else there is no model
If a is true then c must be false or else there is no model…
c is the only way out.

1^{St} Order Logic

Formulas are built by connecting atomic expressions like $c = y+1$ with operators $\wedge$ $\vee$ $\neg$ $\rightarrow$ and quantifiers $\exists$ $\forall$

Notion of satisfiability exists for 1^{st} order formulas

$\exists x\ (\text{person}(x) \wedge \forall y\ ((\text{person}(y) \rightarrow (\text{shaves}(x,y) \leftrightarrow \neg\text{shaves}(y,y))))$

is not satisfiable since y can be x leading to a contradiction

Numerous tools exist for simplifying 1^{st} order formulas to make determining satisfiability faster. These will show up as needed.

1^{St} Order Logic

Let x be a variable and A be a set

Let $P(x)$ be a predicate wrt A if for all values of x,

$P(x)$ is true or false

A is called the *domain* (universe) of discourse and

x is called a *free variable*

Universal Quantifier

The universal quantification of P(x) is the statement:

For all x, P(x) or

For every x, $P(x)$

and is used like this:

$\forall x, P(x)$ or $\forall x \in A, P(x)$

Another possibility: $\forall x, \forall y, P(x, y)$

1^{st} Order Logic

Let x be a variable and A be a set

Let $P(x)$ be a predicate wrt A if for all values of x,

$P(x)$ is true or false

A is called the *domain* (universe) of discourse and

~~x is called a *free variable*~~

Existential Quantifier

The existential quantification of P(x) is the statement:

There exists x, P(x)

and is used like this:

$$\exists x, P(x) \quad \text{or } \exists x \in A, P(x)$$

where variable x is said to be *bound*

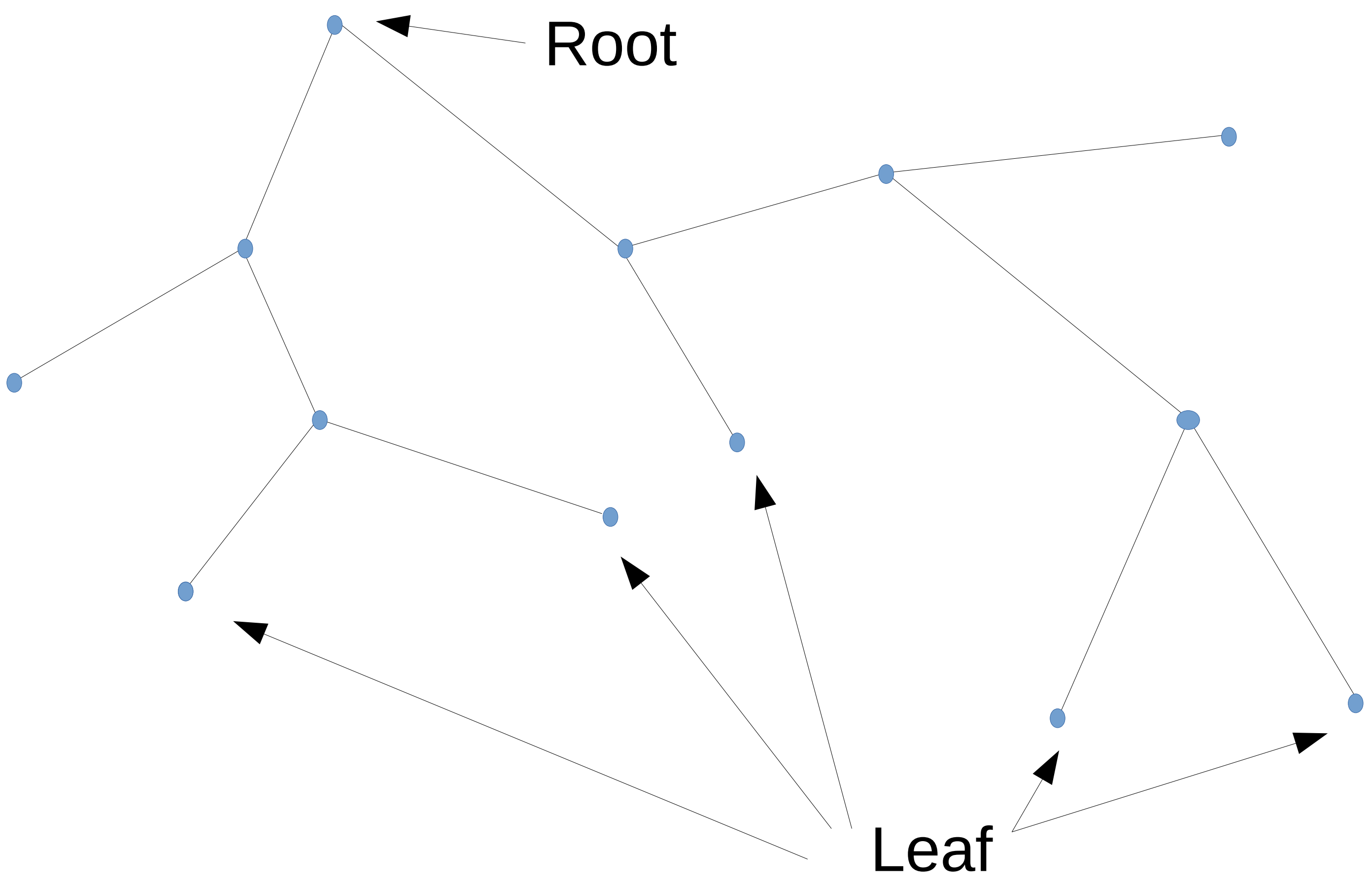
Observe

$$\neg \forall x, P(x) \equiv \exists x, \neg P(x)$$

$$\neg \exists x, P(x) \equiv \forall x, \neg P(x)$$

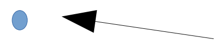
Power of Theorem Provers for 1st Order Logic: Induction

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.



Power of Theorem Provers for 1st Order Logic: Induction

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.

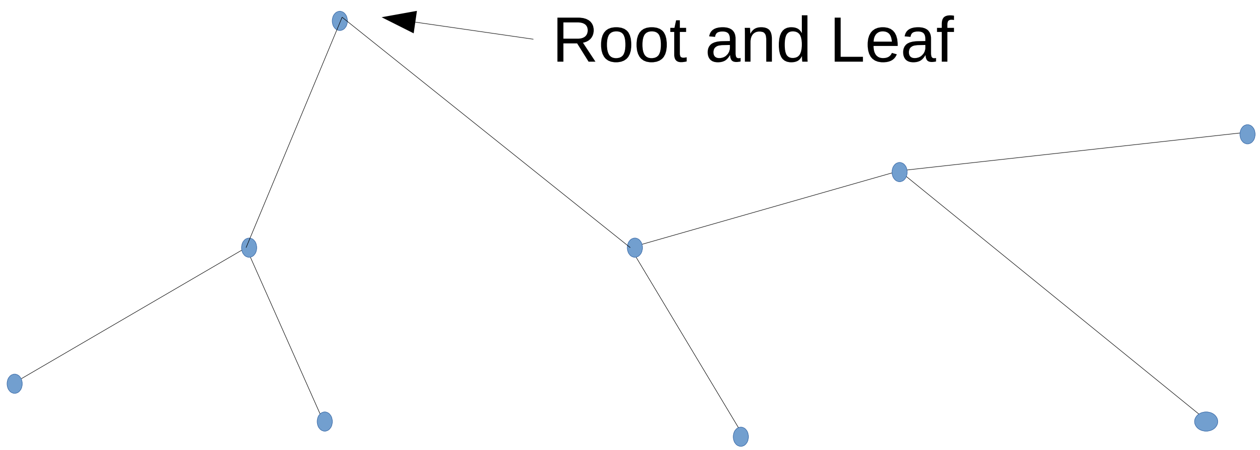


Root and Leaf

Basis: **k = 0:** Just a root – one leaf – (0 + 1 = 1) ✓

Power of Theorem Provers for 1st Order Logic: Induction

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.

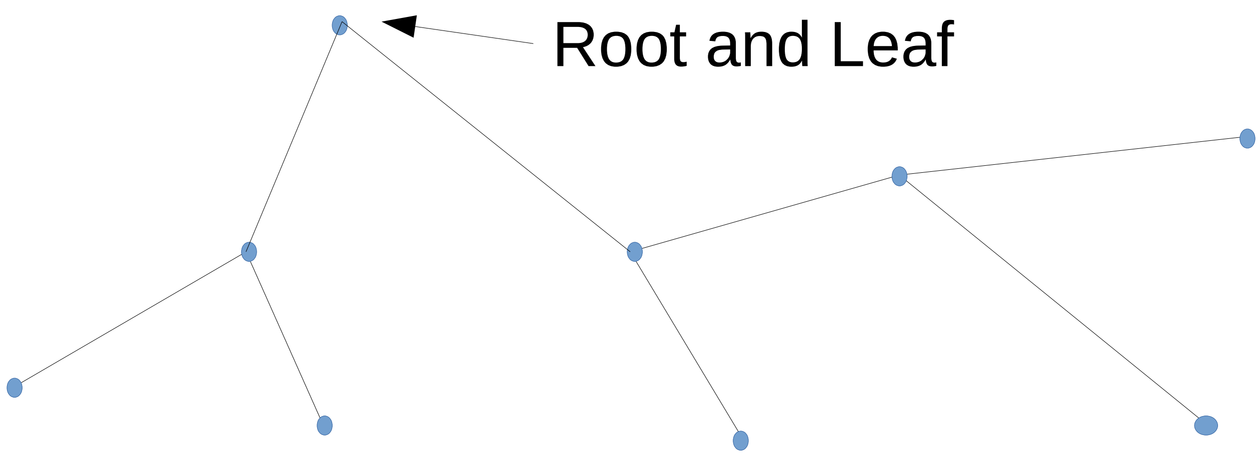


Induction: **k > 0:** Let n be the number of non-leaf nodes
Assume hypothesis correct for n < k
Consider a binary tree with n=k > 0 non-leaf nodes

galois

Power of Theorem Provers for 1st Order Logic: Induction

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.



Induction: **k > 0:** Let n be the number of non-leaf nodes
Assume hypothesis correct for n < k
Consider a binary tree with n=k > 0 non-leaf nodes
Remove one non-leaf node
By hypothesis this tree has (k-1)+1 = k leaves
Since two leaves were removed but one non-leaf node became a leaf, the original tree has k+2-1 = k+1 leaves

Power of Theorem Provers for 1st Order Logic: Induction

Leaves of a binary tree

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.

Representation:

Nested list of pairs or empty lists, called **Nodes**

Node = () | (**Node** **Node**)

the first '**Node**' in (**Node** **Node**) represents the leftmost child of **Node** the second '**Node**' in (**Node** **Node**) represents the rightmost child of **Node**. A **Node** that is () is a leaf.

Example:

((( ) ( )) ((( ) (( ) ( ))) ( )))

Power of Theorem Provers for 1st Order Logic: Induction

Leaves of a binary tree

Prove that the number of leaves of an undirected binary tree is the number of non-leaf nodes plus 1.

Representation:

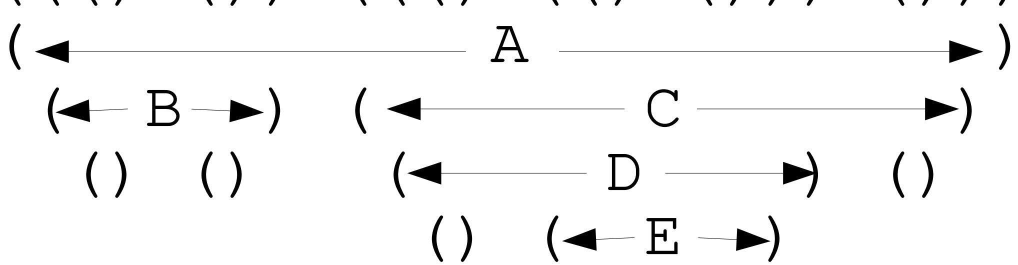
Nested list of pairs or empty lists, called **Nodes**

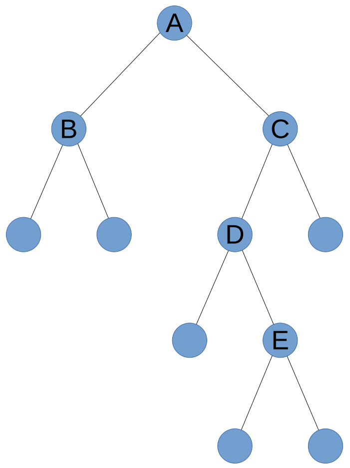
Node = () | (**Node Node**)

the first '**Node**' in (**Node Node**) represents the leftmost child of **Node** the second '**Node**' in (**Node Node**) represents the rightmost child of **Node**. A **Node** that is () is a leaf.

Example:

((( ) ( )) ((( ) (( ) ( ))) ( )))





Power of Theorem Provers for 1st Order Logic: Induction

Leaves of a binary tree

```
;; number of leaves in tree
(defun numb-leaves (tree)
  (if (endp tree)
      1
      (+ (numb-leaves (first tree))
         (numb-leaves (second tree)))))

;; number of internal nodes in tree
(defun numb-internals (tree)
  (if (endp tree)
      0
      (+ (numb-internals (first tree))
         (numb-internals (second tree)) 1)))
```

Power of Theorem Provers for 1st Order Logic: Induction

Leaves of a binary tree

```
;; T iff tree is a binary tree
(defun isABinaryTree (tree)
  (if (endp tree)
      T
      (and (= (len tree) 2)
           (isABinaryTree (first tree))
           (isABinaryTree (second tree)))))

(defthm number-leaves-in-binary-tree
  (implies
    (isABinaryTree tree)
    (= (numb-leaves tree)
       (+ 1 (numb-internals tree)))))
```

✓

Power of Theorem Provers for 1st Order Logic: Induction

Sort a list of comparable objects (strings, numbers, etc.)

Given: algorithm bsort which takes as input a list of comparables
Prove: bsort returns an ordered permutation of the input list

```
(defun bbl (n lst)
   (if (endp lst)
       (list n)
       (if (< n (first lst))
           (cons n lst)
           (cons (first lst) (bbl n (rest lst))))))

;; bubblesort algorithm
(defun bsort (lst)
   (if (endp lst)
       nil
       (bbl (first lst) (bsort (rest lst)))))
```

galois

Power of Theorem Provers for 1st Order Logic: Induction

Sort a list of comparable objects (strings, numbers, etc.)

Given: algorithm bsort which takes as input a list of comparables
Prove: bsort returns an ordered permutation of the input list

```
;; T iff lst is ordered - each element is not
;; greater than the next one in lst
(defun listIsOrdered (lst)
  (or (endp lst)
      (endp (rest lst))
      (and (<= (first lst) (second lst))
           (listIsOrdered (rest lst)))))

;; prove bsort returns an ordered list
(defthm bsort-output-is-ordered
  (listIsOrdered (bsort x)))
```

✓

Power of Theorem Provers for 1st Order Logic: Induction

Sort a list of comparable objects (strings, numbers, etc.)

Given: algorithm bsort which takes as input a list of comparables
Prove: bsort returns an ordered permutation of the input list

```
;; without this ACL2 fails to prove next theorem
;; but ACL2 says why and that info gets theorem
;; below formulated and proved
(defthm bbl-arg-inserted-into-sorted-list
  (member n (bbl n (bsort lst))))

;; perm returns T iff its two args are permutations
(defthm bsort-returns-permutation-of-input
  (perm lst (bsort lst)))

(defthm bsort-sorts-a-given-list
  (let ((sorted-lst (bsort lst)))
    (and (listIsOrdered sorted-lst)
         (perm lst sorted-lst))))
```

Power of Theorem Provers for 1st Order Logic: Induction

Pigeon Hole Principle

Given: n pigeon holes and more than n pigeons

Prove: if all pigeons are assigned holes, at least one hole must have at least two pigeons

Power of Theorem Provers for 1st Order Logic: Induction

Pigeon Hole Principle

Given: n pigeon holes and more than n pigeons
Prove: if all pigeons are assigned holes, at least one hole must have at least two pigeons

Representation:

Vector of non-negative integers, one per hole
Each number is the number of pigeons in the hole

Total number of pigeons = sum of numbers in the vector
Total number of holes = length of the vector

galois

Power of Theorem Provers for 1st Order Logic: Induction

Total number of pigeons = sum of numbers in the vector

```
(defun sum-list (l)
  (if (endp l)
      0
      (+ (first l) (sum-list (rest l)))))
```

true iff list l has 0 and 1 elements only

```
(defun posn-one-listp (l)
  (if (endp l)
      T
      (and (or (= (first l) 0) (= (first l) 1))
           (posn-one-listp (rest l)))))
```

Theorem:

```
(defthm pigeon-hole
  (implies
    (and (< 0 (len l)) (< (len l) (sum-list l))
    (not (posn-one-listp l))))
```

✓