Logical Algorithms

Tamás Mihálydeák
mihalydeak.tamas@inf.unideb.hu

Department of Computer Science

May 6, 2019

The main task of logic

- to give the laws of valid arguments (inferences, consequence relations)

Valid arguments

- Valid arguments (inferences):
  - an argument (an inference): a relation between premise(s) and conclusion
  - a consequence relation
    - input: premise(s)
    - output: conclusion
  - Valid arguments (inferences, consequence relations): if all premises are true, then the conclusion is true.
  - Logically valid arguments: when the former holds necessarily.

Definition/1

Classical zero–order language is an ordered triple

$$L^{(0)} = \langle LC, Con, Form \rangle$$

where

1. $LC = \{\neg, \supset, \wedge, \vee, \equiv, (,)\}$ (the set of logical constants).
2. $Con \neq \emptyset$ the countable set of non-logical constants (propositional parameters)
3. $LC \cap Con = \emptyset$
4. The set of formulae i.e. the set $Form$ is given by the following inductive definition:

Definition/2

- $Con \subseteq Form$
- If $A \in Form$, then $\neg A \in Form$.
- If $A, B \in Form$, then
  - $(A \supset B) \in Form$,
  - $(A \wedge B) \in Form$,
  - $(A \vee B) \in Form$,
  - $(A \equiv B) \in Form$.

Remark

The members of the set Con are the atomic formulae (prime formulae).

Definition

- If A is an atomic formula, then it has no direct subformula;
- $\neg A$ has exactly one direct subformula: A;
- Direct subformulae of formulae $(A \supset B)$, $(A \wedge B)$, $(A \vee B)$, $(A \equiv B)$ are formulae A and B, respectively.

Definition

The set of subformulae of formula A [denoting: $SF(A)$] is given by the following inductive definition:

1. $A \in SF(A)$ (i.e. the formula A is a subformula of itself);
2. if $A' \in SF(A)$ and B is a direct subformula of A'-nek, then $B \in SF(A)$ (i.e., if A' is a subformula of A, then all direct subformulae of A' are subformulae of A).

Definition

The contruction tree of a formula A is a finite ordered tree whose nodes are formulae,

- the root of the tree is the formmula A,
- the node with formula $\neg B$ has one child: he node with the formula B,
- the node with formulae $(B \supset C)$, $(B \wedge C)$, $(B \vee C)$, $(B \equiv C)$ has two children: the nodes with B, and C
- the leaves of the tree are atomic formulae.

Definition

The function ϱ is an interpretation of the language $L^{(0)}$ if

1. $Dom(\varrho) = Con$
2. If $p \in Con$, then $\varrho(p) \in \{0, 1\}$.

Definition

Let ϱ be an interpretation and $|A|_\varrho$ be the semantic value of the formula A formula with respect to ϱ.

1. If $p \in Con$, then $|p|_\varrho = \varrho(p)$
2. If $A \in Form$, then $|\neg A|_\varrho = 1 - |A|_\varrho$.
3. If $A, B \in Form$, then
   - $|(A \supset B)|_\varrho = \begin{cases} 0 & \text{if } |A|_\varrho = 1, \text{ and } |B|_\varrho = 0; \\ 1, & \text{otherwise} \end{cases}$
   - $|(A \wedge B)|_\varrho = \begin{cases} 1 & \text{if } |A|_\varrho = 1, \text{ and } |B|_\varrho = 1; \\ 0, & \text{otherwise} \end{cases}$
   - $|(A \vee B)|_\varrho = \begin{cases} 0 & \text{if } |A|_\varrho = 0, \text{ and } |B|_\varrho = 0; \\ 1, & \text{otherwise.} \end{cases}$
   - $|(A \equiv B)|_\varrho = \begin{cases} 1 & \text{if } |A|_\varrho = |B|_\varrho; \\ 0, & \text{otherwise.} \end{cases}$

Definition (model – a set of formulas)

Let $\Gamma \subseteq Form$ be a set of formulas. An interpretation ϱ is a model of the set of formulas Γ, if $|A|_\varrho = 1$ for all $A \in \Gamma$.

Definition – a model of a formula

A model of a formula A is the model of the singleton $\{A\}$.

Definition – satisfiable a set of formulas

The set of formulas $\Gamma \subseteq Form$ is satisfiable if it has a model.
(If there is an interpretation in which all members of the set Γ are ture.)

Definition – satisfiable a formula

A formula $A \in Form$ is satisfiable, if the singleton $\{A\}$ is satisfiable.

Remark

- A satisfiable set of formulas does not involve a logical contradiction; its formulas may be true together.
- A safisfiable formula may be true.
- If a set of formulas is satisfiable, then its members are satisfiable.
- But: all members of the set $\{p, \neg p\}$ are satisfiable, and the set is not satisfiable.

Theorem

All subsets of a satisfiable set are satisfiable.

Proof

- Let $\Gamma \subseteq Form$ be a set of formulas and $\Delta \subseteq \Gamma$.
- Γ is satisfiable: it has a model. Let ϱ be a model of Γ.
- A property of ϱ: If $A \in \Gamma$, then $|A|_\varrho = 1$
- Since $\Delta \subseteq \Gamma$, if $A \in \Delta$, then $A \in \Gamma$, and so $|A|_\varrho = 1$. That is the interpretation ϱ is a model of Δ, and so Δ is satisfiable.

Definition – unsatisfiable set

The set $\Gamma \subseteq Form$ is unsatisfiable if it is not satisfiable.

Definition – unsatisfiable formula

A formula $A \in Form$ is unsatisfiable if the singleton $\{A\}$ is unsatisfiable.

Remark

A unsatisfiable set of formulas involve a logical contradiction. (Its members cannot be true together.)

Theorem

All expansions of an unsatisfiable set of formulas are unsatisfiable.

Indirect proof

- Suppose that $\Gamma \subseteq Form$ is an unsatisfiable set of formulas and $\Delta \subseteq Form$ is a set of formulas.
- Indirect condition: Γ is unsatisfiable, and $\Gamma \cup \Delta$ satisfiable.
- $\Gamma \subseteq \Gamma \cup \Delta$
- According to the former theorem Γ is satisfiable, and it is a contradiction.

Definition

A formula A is the logical consequence of the set of formulas Γ if the set $\Gamma \cup \{\neg A\}$ is unsatifiable. $(Notation : \Gamma \vDash A)$

Definition

$A \vDash B$, if $\{A\} \vDash B$.

Definition

The formula A is valid if $\emptyset \vDash A$. (Notation: $\vDash A$)

The formulas A and B are logically equivalent if $A \vDash B$ and $B \vDash A$. (Notation: $A \Leftrightarrow B$)

Theorem

Let $\Gamma \subseteq Form$, and $A \in Form$. $\Gamma \vDash A$ if and only if all models of the set Γ are the models of formula A. (i.e. the singleton $\{A\}$).

Proof

$\rightarrow$ Indirect condition: There is a model of $\Gamma \vDash A$ such that it is not a model of the formula A.
Let the interpretation ϱ be this model.
The properties of ϱ:

1. $|B|_\varrho = 1$ for all $B \in \Gamma$;
2. $|A|_\varrho = 0$, and so $|\neg A|_\varrho = 1$

In this case all members of the set $\Gamma \cup \{\neg A\}$ are true wrt ϱ-ban, and so $\Gamma \cup \{\neg A\}$ is satisfiable. It means that $\Gamma \nvDash A$, and it is a contradiction.

Proof

$\leftarrow$ Indirect condition: All models of the set Γ are the models of formula A, but (and) $\Gamma \nvDash A$.
In this case $\Gamma \cup \{\neg A\}$ is satisfiable, i.e. it has a model.
Let the interpretation ϱ be a model.
The properties of ϱ:

1. $|B|_\varrho = 1$ for all $B \in \Gamma$;
2. $|\neg A|_\varrho = 1$, i.e. $|A|_\varrho = 0$

So the set Γ has a model such that it is not a model of formula A, and it is a contradiction.

Corollary

Let $\Gamma \subseteq Form$, and $A \in Form$. $\Gamma \vDash A$ if and only if for all interpretations in which all members of Γ are true, the formula A is true.

Theorem

If A is a valid formula $((\vDash A))$, then $\Gamma \vDash A$ for all sets of formulas Γ. (A valid formula is a consequence of any set of formulas.)

Proof

- If A is a valid formula, then $\emptyset \vDash A$ (according to its definition).
- $\emptyset \cup \{\neg A\}$ $(= \{\neg A\})$ is unsatisfiable, and so its expansions are unsatisfiable.
- $\Gamma \cup \{\neg A\}$ is an expansion of $\{\neg A\}$, and so it is unsatisfiable, i.e. $\Gamma \vDash A$.

Theorem

If Γ is unsatisfiable, then $\Gamma \vDash A$ for all A. (All formulas are the consequences of an unsatisfiable set of formulas.)

Proof

- According to a proved theorem: If Γ is unsatisfiable, the all expansions of Γ are unsatisfiable.
- $\Gamma \cup \{\neg A\}$ is an expansion of Γ, and so it is unsatisfiable, i.e. $\Gamma \vDash A$.

Theorem

Deduction theorem: If $\Gamma \cup \{A\} \vDash B$, then $\Gamma \vDash (A \supset B)$.

Proof

- Indirect condition: Suppose, that $\Gamma \cup \{A\} \vDash B$, and $\Gamma \nvDash (A \supset B)$.
- $\Gamma \cup \{\neg(A \supset B)\}$ is satisfiable, and so it has a model. Let the interpretation ϱ be a model.
- The properties of ϱ:
  1. All members of Γ are true wrt ϱ.
  2. $|\neg(A \supset B)|_\varrho = 1$
- $|(A \supset B)|_\varrho = 0$, i.e. $|A|_\varrho = 1$ and $|B|_\varrho = 0$. So$|\neg B|_\varrho = 1$.
- All members of $\Gamma \cup \{A\} \cup \{\neg B\}$ are true wrt interpretation ϱ, i.e. $\Gamma \cup \{A\} \nvDash B$, and it is a contradiction.

Theorem

In the opposite direction: If $\Gamma \vDash (A \supset B)$, then $\Gamma \cup \{A\} \vDash B$.

Proof

- Indirect condition: Suppose that $\Gamma \vDash (A \supset B)$, and $\Gamma \cup \{A\} \nvDash B$.
- So $\Gamma \cup \{A\} \cup \{\neg B\}$ is satisfiable, i.e. it has a model. Let the interpretation ϱ a model.
- The properties of ϱ:
  1. All members of Γ are true wrt the interpretation ϱ.
  2. $|A|_\varrho = 1$
  3. $|\neg B|_\varrho = 1$, and so $|B|_\varrho = 0$
- $|(A \supset B)|_\varrho = 0$, $|\neg(A \supset B)|_\varrho = 1$.
- All members of $\Gamma \cup \{\neg(A \supset B)\}$ are true wrt the interpretation ϱ, i.e. $\Gamma \nvDash (A \supset B)$.

Corollary

$A \vDash B$ if and only if $\vDash (A \supset B)$

Proof

Let $\Gamma = \emptyset$ in the former theorems.

Cut elimination theorem

If $\Gamma \cup \{A\} \vDash B$ and $\Delta \vDash A$, then $\Gamma \cup \Delta \vDash B$.

Proof

Indirect.

Problems

- Is there any algorithm to decide the satisfiability of a given formula? This problem is crucial:
  - Finite set of formulas $\to$ formula;
  - Consequence relation with finite set of premisses $\to$ formula

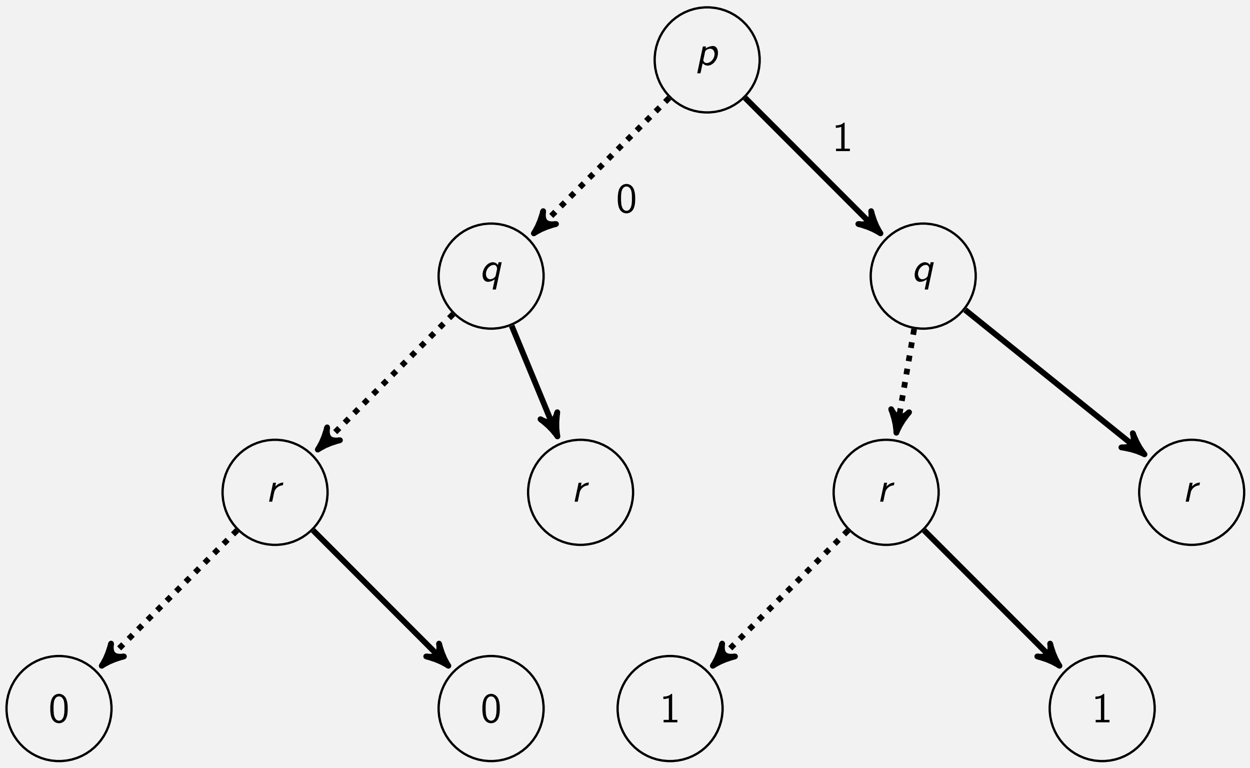
Truth tables

	p	q	r	$p \vee (q \wedge r)$
1.	1	1	1	1
2.	1	1	0	1
3.	1	0	1	1
4.	1	0	0	1
5.	0	1	1	1
6.	0	1	0	0
7.	0	0	1	0
8.	0	0	0	0

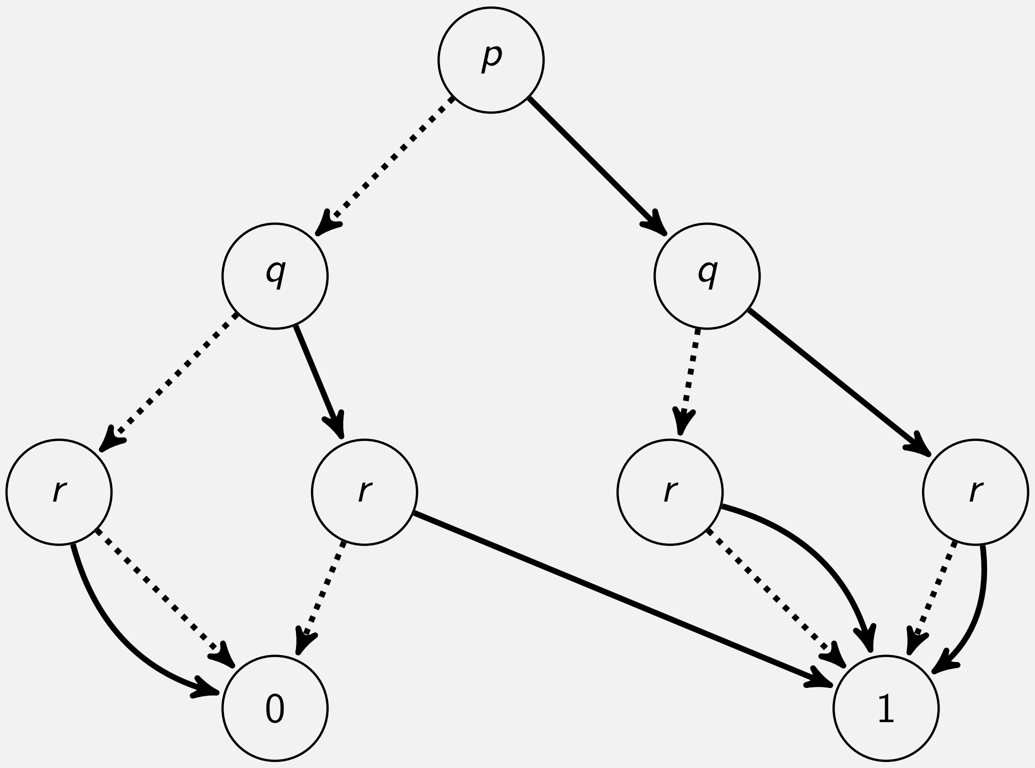
	p	q	r	$p \vee (q \wedge r)$
1.,2.	1	1	$\star$	1
3.,4.	1	0	$\star$	1
5.	0	1	1	1
6.	0	1	0	0
7.,8.	0	0	$\star$	0

	p	q	r	$p \vee (q \wedge r)$
1.,2., 3.,4.	1	$\star$	$\star$	1
5.	0	1	1	1
6.	0	1	0	0
7.,8.	0	0	$\star$	0

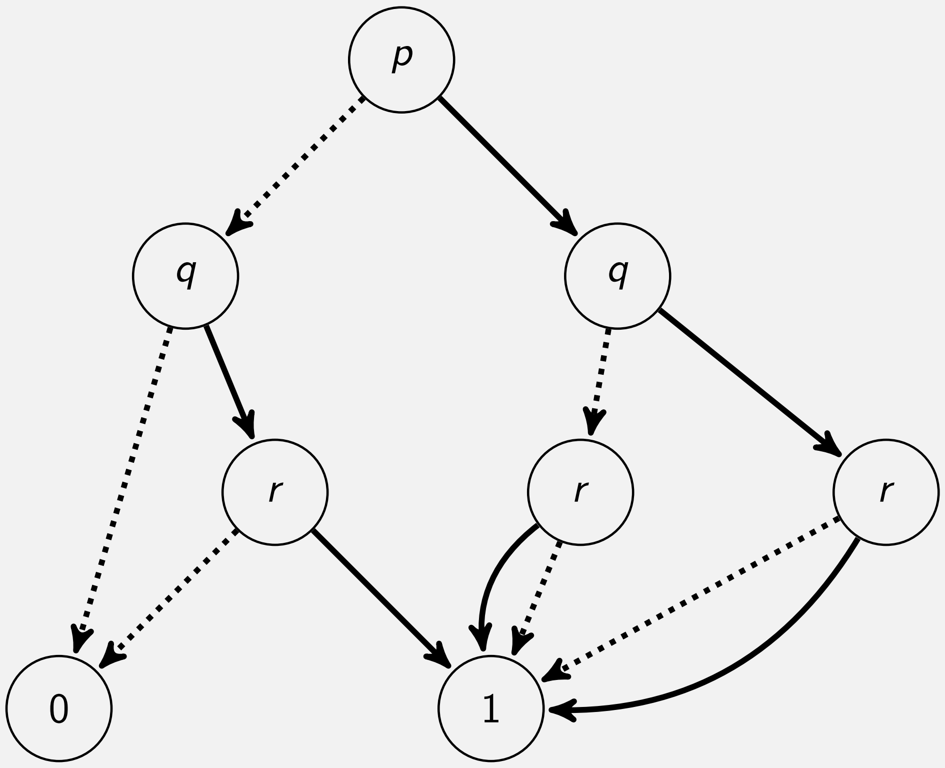
$p \vee (q \wedge r)$



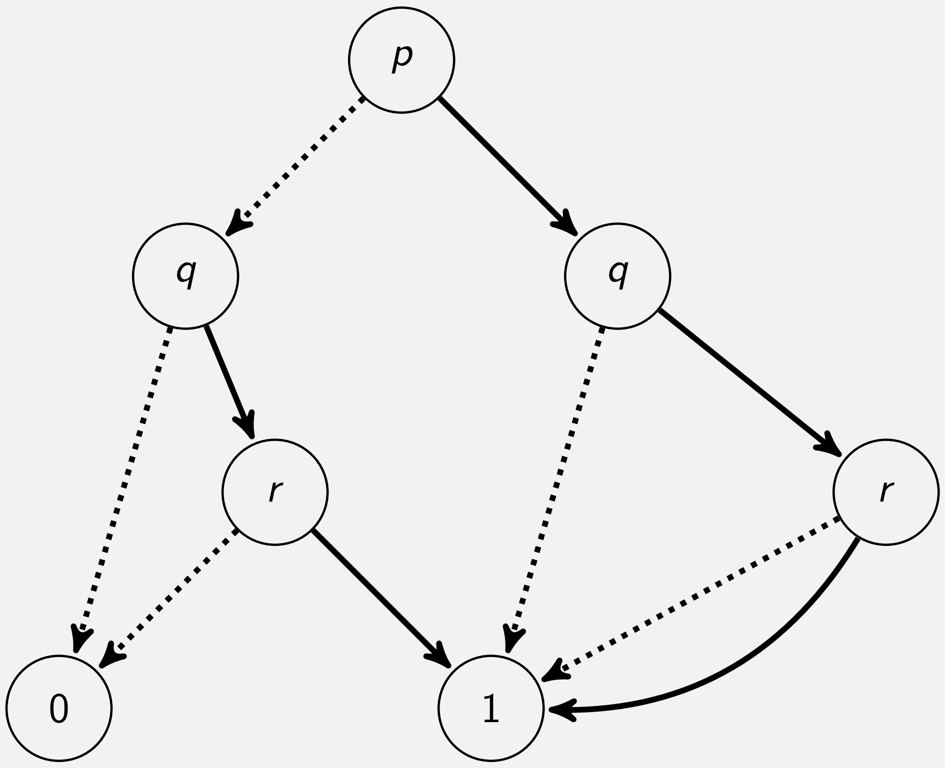
$p \vee (q \wedge r)$



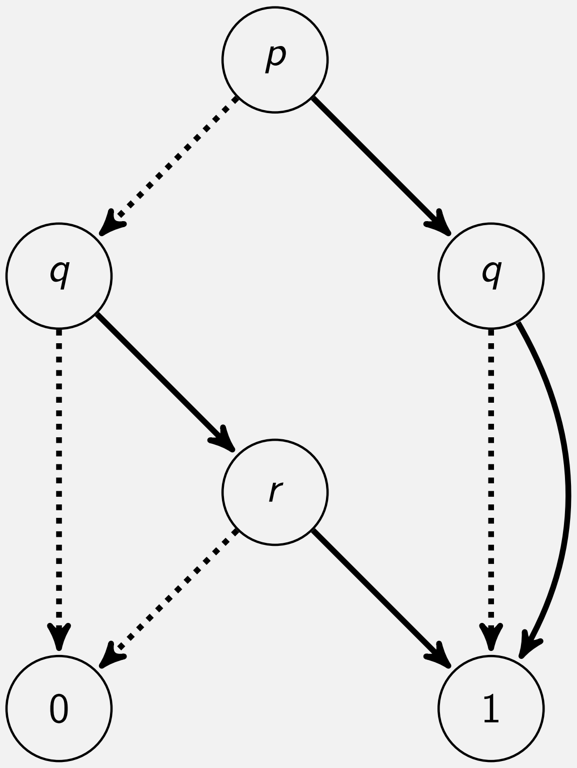
$p \vee (q \wedge r)$



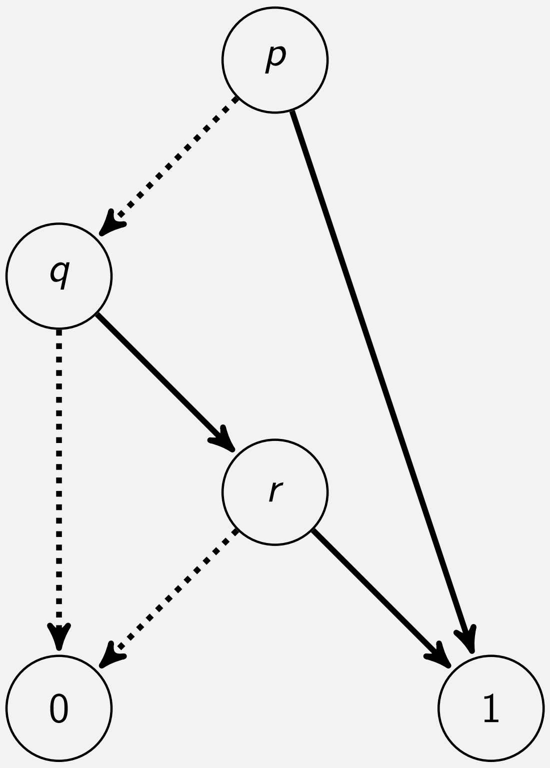
$p \vee (q \wedge r)$



$p \vee (q \wedge r)$



$p \vee (q \wedge r)$



Binary decision diagram (BDD)

- A binary decision diagram (BDD) is a data structure for representing the semantics of a formula in propositional logic:
  - formula $\to$ directed graph $\to$ algorithm $\to$ directed reduced graph;
  - reduced graphs have the property that the graphs for logically equivalent formulas are identical;
  - A formula is valid iff its reduced BDD is identical to the trivial BDD for true (i.e. the trivial BDD of verum $[\uparrow \Leftrightarrow_{def} (p \vee \neg p)]$);
  - a formula is satisfiable iff its reduced BDD is not identical to the trivial BDD for false (i.e. the trivial BDD of falsum $[\downarrow \Leftrightarrow_{def} (p \wedge \neg p)]$).

Definition

Let $A \in Form$ (i.e. A is a formula of propositional logic). A binary decision diagram (BDD) for a formula A in propositional logic is a directed acyclic graph such that

- each leaf is labeled with a truth value 1 or 0;
- each interior node is labeled with a parameter and has two outgoing edges: one, the false edge, is denoted by a dotted line, while the other, the true edge, is denoted by a solid line;
- no atom appears more than once in a branch from the root to an edge.

Remark

- Given a branch b and its associated interpretation ρ, the leaf is labeled with $|A|_{\varrho}$, the truth value of the formula wrt ρ.
- If the interpretation is partial, it must assign to enough atoms so that the truth value is defined.

Algorithm Reduce

Input: A binary decision diagram *bdd*.
Output: A reduced binary decision diagram *bdd*′.

- If *bdd* has more than two distinct leaves (one labeled 1 and one labeled 0), remove duplicate leaves. Direct all edges that pointed to leaves to the remaining two leaves.
- Perform the following steps as long as possible:
    1. If both outgoing edges of a node labeled p_i point to the same node labeled p_j , delete this node for p_i and direct p_i's incoming edges to p_j.
    2. If two nodes labeled p_i are the roots of identical sub-BDDs, delete one sub- BDD and direct its incoming edges to the other node.

Definition

A BDD that results from applying the algorithm Reduce is a reduced binary decision diagram.

Theorem

The reduced BDD *bdd*′ returned by the algorithm Reduce is logically equivalent to the input BDD *bdd* (in the sense that it determines the same truth value in all interpretations).

The truth table of negation

$\neg$	$\neg p$
0	1
1	0

- The law of double negation: $\neg\neg A \Leftrightarrow A$

The truth table of conjunction

$\wedge$	0	1	(q)
0	0	0	
1	0	1	
(p)			

- Commutative: $(A \wedge B) \Leftrightarrow (B \wedge A)$ for all $A, B \in Form$.
- Associative: $(A \wedge (B \wedge C)) \Leftrightarrow ((A \wedge B) \wedge C)$ for all $A, B, C \in Form$.
- Idempotent: $(A \wedge A) \Leftrightarrow A$ for all $A \in Form$.

- $(A \wedge B) \vDash A$, $(A \wedge B) \vDash B$
- The law of contradiction: $\vDash \neg(A \wedge \neg A)$
- The set $\{A_1, A_2, \dots, A_n\}$ $(A_1, A_2, \dots, A_n \in Form)$ is satisfiable iff the formula $A_1 \wedge A_2 \wedge \cdots \wedge A_n$ is satisfiable.
- The set $\{A_1, A_2, \dots, A_n\}$ $(A_1, A_2, \dots, A_n \in Form)$ is unsatisfiable iff the formula $A_1 \wedge A_2 \wedge \cdots \wedge A_n$ is unsatisfiable.
- $\{A_1, A_2, \dots, A_n\} \vDash A$ $(A_1, A_2, \dots, A_n, A \in Form)$ iff $A_1 \wedge A_2 \wedge \cdots \wedge A_n \vDash A$.
- $\{A_1, A_2, \dots, A_n\} \vDash A$ $(A_1, A_2, \dots, A_n, A \in Form)$ iff the formula $((A_1 \wedge A_2 \wedge \cdots \wedge A_n) \wedge \neg A)$ is unsatisfiable.

The truth table of disjunction:

$\vee$	0	1
0	0	1
1	1	1

- Commutative: $(A \vee B) \Leftrightarrow (B \vee A)$ for all $A, B \in Form$.
- Associative: $(A \vee (B \vee C)) \Leftrightarrow ((A \vee B) \vee C)$ for all $A, B, C \in Form$.
- Idempotent: $(A \vee A) \Leftrightarrow A$ for all $A \in Form$.
- $A \vDash (A \vee B)$ for all $A, B \in Form$.
- $\{(A \vee B), \neg A\} \vDash B$
- The law of excluded middle: $\vDash (A \vee \neg A)$

- Connection between conjunction and disjunction:

$\wedge$	0	1
0	0	0
1	0	1

	1	0
1	1	1
0	1	0

$\vee$	0	1
0	0	1
1	1	1

- Conjunction and disjunction are dual truth functors.

- Two laws of distributivity:
  - $(A \vee (B \wedge C)) \Leftrightarrow ((A \vee B) \wedge (A \vee C))$
  - $(A \wedge (B \vee C)) \Leftrightarrow ((A \wedge B) \vee (A \wedge C))$

- Properties of absorption
  - $(A \wedge (B \vee A)) \Leftrightarrow A$
  - $(A \vee (B \wedge A)) \Leftrightarrow A$

De Morgan's laws

- What do we say when we deny a conjunction?
- What do we say when we deny a disjunction?
- $\neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B)$
- $\neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$

- The proofs of De Morgan's laws.
-

A	B	$\neg A$	$\neg B$	$(\neg A \wedge \neg B)$	$(A \vee B)$	$\neg(A \vee B)$
0	0	1	1	1	0	1
0	1	1	0	0	1	0
1	0	0	1	0	1	0
1	1	0	0	0	1	0

- The truth table of implication:

$\supset$	0	1
0	1	1
1	0	1

- $\vDash (A \supset A)$
- Modus ponens: $\{(A \supset B), A\} \vDash B$
- Modus tollens:
  $\{(A \supset B), \neg B\} \vDash \neg A$
- Chain rule: $\{(A \supset B), (B \supset C)\} \vDash (A \supset C)$
- Reduction to absurdity: $\{(A \supset B), (A \supset \neg B)\} \vDash \neg A$

- $\neg A \vDash (A \supset B)$
- $B \vDash (A \supset B)$
- $((A \wedge B) \supset C) \Leftrightarrow (A \supset (B \supset C))$
- Contraposition: $(A \supset B) \Leftrightarrow (\neg B \supset \neg A)$
- $(A \supset \neg A) \vDash \neg A$
- $(\neg A \supset A) \vDash A$
- $(A \supset (B \supset C)) \Leftrightarrow ((A \supset B) \supset (A \supset C))$
- $\vDash (A \supset (\neg A \supset B))$
- $((A \vee B) \supset C) \Leftrightarrow ((A \supset C) \wedge (B \supset C))$
- $\{A_1, A_2, \ldots, A_n\} \vDash A$ ($A_1, A_2, \ldots, A_n, A \in Form$) iff the formula $((A_1 \wedge A_2 \wedge \cdots \wedge A_n) \supset A)$ is valid.

- The truth table of (material) equivalence:

$\equiv$	0	1
0	1	0
1	0	1

- $\models (A \equiv A)$
- $\models \neg(A \equiv \neg A)$

Expressibility

- $(A \supset B) \Leftrightarrow \neg(A \wedge \neg B)$
- $(A \supset B) \Leftrightarrow (\neg A \vee B)$
- $(A \wedge B) \Leftrightarrow \neg(A \supset \neg B)$
- $(A \vee B) \Leftrightarrow (\neg A \supset B)$
- $(A \vee B) \Leftrightarrow \neg(\neg A \wedge \neg B)$
- $(A \wedge B) \Leftrightarrow \neg(\neg A \vee \neg B)$
- $(A \equiv B) \Leftrightarrow ((A \supset B) \wedge (B \supset A))$

Theory of truth functors

Base

- A base is a set of truth functors whose members can express all truth functors.
  - For example: $\{\neg, \supset\}$,$\{\neg, \wedge\}$, $\{\neg, \vee\}$
    1. $(p \wedge q) \Leftrightarrow \neg(p \supset \neg q)$
    2. $(p \vee q) \Leftrightarrow (\neg p \supset q)$
  - Truth functor Sheffer: $(p|q) \Leftrightarrow_{def} \neg(p \wedge q)$
  - Truth functor neither-nor: $(p \parallel q) \Leftrightarrow_{def} (\neg p \wedge \neg q)$
  - Remark: Singleton bases: $(p|q)$, $(p \parallel q)$

Definition

If $p \in Con$, then formulas $p, \neg p$ are literals (p is the base of the literals).

Definition

If the formula A is a literal or a conjunction of literals, then A is an elementary conjunction.

Definition

If the formula A is a literal or a disjunction of literals, the A is an elementary disjunction.

Remark

If the literals of an elementary conjunction/disjunction have different bases, then the elementary conjunction/disjunction represents an interpretation (or a family of interpretations).

Definition

A disjunction of elementary conjunctions is a disjunctive normal form.

Definition

A conjunction of elementary disjunctions is a conjunctive normal form.

Theorem

There is a normal form of any formula of proposition logic, i. e. if $A \in Form$, then there is a formula B such that B is a normal form and $A \Leftrightarrow B$

Definition

Let $L^{(0)} = \langle LC, Con, Form \rangle$ be a language of classical propositional logic and $(LC = \{\neg, \supset, (,)\})$.
The axiom scheme of classical propositional calculus:

- (A1): $A \supset (B \supset A)$
- (A2): $(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C))$
- (A3): $(\neg A \supset \neg B) \supset (B \supset A)$

Definition

- The regular substitution of axiom schemes are formulas, such that A, B, C are replaced by arbitrary formulas.
- The axioms of classical propositional calculus are the regular substitutions of axiom schemes.

The inductive definition of syntactical consequence relation

- Let $\Gamma \subseteq Form, A \in Form$. The formula A is a syntactical consequence of the set Γ (in noation $\Gamma \vdash A$), if at least one of the followings holds:
  1. if $A \in \Gamma$, then $\Gamma \vdash A$;
  2. if A is an axiom, then $\Gamma \vdash A$;
  3. if $\Gamma \vdash B$, and $\Gamma \vdash B \supset A$, then $\Gamma \vdash A$.

Definition

Let $\Gamma \subset Form, A \in Form$. If formula A is a syntactical consequence of the set Γ, then '$\Gamma \vdash A$' is a sequence.

The fundamental rule of natural deduction is based on deduction theorem.

Deduction theorem

Ifa $\Gamma \cup \{A\} \vdash B$, then $\Gamma \vdash A \supset B$.

Deduction theorem can be written in the following form:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}$$

Structural rules/1

In the following let $\Gamma, \Delta \subseteq Form$, $A, B, C, \in Form$.

Rule of assumption

$$\frac{\emptyset}{\Gamma, A \vdash A}$$

Rule of expansion

$$\frac{\Gamma \vdash A}{\Gamma, B \vdash A}$$

Rule of constriction

$$\frac{\Gamma, B, B, \Delta \vdash A}{\Gamma, B, \Delta \vdash A}$$

Structural rules/2

Rule of permutation

$$\frac{\Gamma, B, C, \Delta \vdash A}{\Gamma, C, B, \Delta \vdash A}$$

Cut rule

$$\frac{\Gamma \vdash A \qquad \Delta, A \vdash B}{\Gamma, \Delta \vdash B}$$

Logical rules/1

Rules of implication (introduction and elimination)

$$(\supset\ 1.)\quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B} \qquad\qquad (\supset\ 2.)\quad \frac{\Gamma \vdash A \qquad \Gamma \vdash A \supset B}{\Gamma \vdash B}$$

Rules of conjunction

$$(\wedge\ 1.)\quad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad\qquad (\wedge\ 2.)\quad \frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}$$

Rules of disjunction

$$(\vee\ 1.)\quad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \qquad\qquad (\vee\ 2.)\quad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$$

$$(\vee\ 3.)\quad \frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}$$

Logical rules/2

Rules of negation

$$(\neg\ 1.)\quad \frac{\Gamma, A \vdash B \qquad \Gamma, A \vdash \neg B}{\Gamma \vdash \neg A} \qquad\qquad (\neg\ 2.)\quad \frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A}$$

Rules of material equivalence

$$(\equiv\ 1.)\quad \frac{\Gamma, A \vdash B \qquad \Gamma, B \vdash A}{\Gamma \vdash A \equiv B}$$

$$(\equiv\ 2.)\quad \frac{\Gamma \vdash A \qquad \Gamma \vdash A \equiv B}{\Gamma \vdash B}$$

$$(\equiv\ 3.)\quad \frac{\Gamma \vdash B \qquad \Gamma \vdash A \equiv B}{\Gamma \vdash A}$$

Examples

$$\frac{\Gamma, A \vdash B}{\Gamma, \neg B \vdash \neg A} \tag{1}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\Gamma, A \vdash B}{\Gamma, A, \neg B \vdash B}\,(\text{Expansion})}{\Gamma, \neg B, A \vdash B}\,(\text{Permutation}) \qquad \dfrac{\dfrac{\emptyset}{\Gamma, A, \neg B \vdash \neg B}\,(\text{Assumption})}{\Gamma, \neg B, A \vdash \neg B}\,(\text{Permutation})}{\Gamma, \neg B \vdash \neg A}\,(\neg\ 1.)$$

Examples

$$\frac{\Gamma, A \vdash \neg B}{\Gamma, B \vdash \neg A} \tag{2}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\emptyset}{\Gamma, A, B \vdash B}\,(\text{Asumption})}{\Gamma, B, A \vdash B}\,(\text{Permutation}) \qquad \dfrac{\dfrac{\Gamma, A \vdash \neg B}{\Gamma, A, B \vdash \neg B}\,(\text{Expansion})}{\Gamma, B, A \vdash \neg B}\,(\text{Permutation})}{\Gamma, B \vdash \neg A}\,(\neg\ 1.)$$

Examples

$$\frac{\Gamma, \neg A \vdash B}{\Gamma, \neg B \vdash A} \tag{3}$$

Proof:

$$\begin{array}{r}(\text{Expansion})\\(\text{Permutation})\\(\neg\ 1.)\end{array}\;
\dfrac{\dfrac{\dfrac{\Gamma, \neg A \vdash B}{\Gamma, \neg A, \neg B \vdash B}}{\Gamma, \neg B, \neg A \vdash B} \qquad \dfrac{\dfrac{\emptyset}{\Gamma, \neg A, \neg B \vdash \neg B}}{\Gamma, \neg B, \neg A \vdash \neg B}}{(\neg\ 2.)\ \dfrac{\Gamma, \neg B \vdash \neg\neg A}{\Gamma, \neg B \vdash A}}
\;\begin{array}{l}(\text{Assumption})\\(\text{Permutation})\\ \end{array}$$

Examples

$$\frac{\Gamma, \neg A \vdash \neg B}{\Gamma, B \vdash A} \tag{4}$$

Proof:

$$\begin{array}{r}(\text{Asumption})\\(\text{Permutation})\\(\neg\ 1.)\end{array}\;
\dfrac{\dfrac{\dfrac{\emptyset}{\Gamma, \neg A, B \vdash B}}{\Gamma, B, \neg A \vdash B} \qquad \dfrac{\dfrac{\Gamma, \neg A \vdash \neg B}{\Gamma, \neg A, B \vdash \neg B}}{\Gamma, B, \neg A \vdash \neg B}}{(\neg\ 2.)\ \dfrac{\Gamma, B \vdash \neg\neg A}{\Gamma, B \vdash A}}
\;\begin{array}{l}(\text{Expansion})\\(\text{Permutation})\\ \end{array}$$

Examples

$$\vdash A \supset A \tag{5}$$

Proof:

$$\begin{array}{rc} & \emptyset \\ \cline{2-2} \text{(Assumption)} & A \vdash A \\ \cline{2-2} (\supset 1.) & \vdash A \supset A \end{array}$$

Examples

$$A, A \supset B \vdash B \tag{6}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\emptyset}{A \supset B, A \vdash A}}{A,\ A \supset B \vdash A} \qquad \dfrac{\emptyset}{A, A \supset B \vdash A \supset B}}{A, A \supset B \vdash B}$$

Examples

$$A \vdash B \supset A \tag{7}$$

Proof:

$$\text{(}\supset\text{ 1.)}\ \dfrac{\text{(Permutation)}\ \dfrac{\text{(Assumption)}\ \dfrac{\emptyset}{B, A \vdash A}}{A, B \vdash A}}{A \vdash B \supset A}$$

Examples

$$A, \neg A \vdash B \tag{8}$$

$$\neg A \vdash A \supset B \tag{9}$$

Proof (8), (9):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A, \neg B, \neg A \vdash \neg A}}{A, \neg A, \neg B \vdash \neg A} \qquad \dfrac{\dfrac{\dfrac{\emptyset}{\neg A, \neg B, A \vdash A}}{\neg A, A, \neg B \vdash A}}{A, \neg A, \neg B \vdash A}}{A, \neg A \vdash \neg\neg B}}{A, \neg A \vdash B}}{\neg A, A \vdash B}}{\neg A \vdash A \supset B}$$

Examples

$$B \vdash A \supset B \tag{10}$$

Proof:

$$\cfrac{\cfrac{\cfrac{\emptyset}{B \vdash B}}{B, A \vdash B}}{B \vdash A \supset B}$$

Examples

$$\vdash A \supset B \equiv \neg A \vee B \tag{11}$$

Proof: At first let us prove that

$$A \supset B \vdash \neg A \vee B \tag{12}$$

$$\cfrac{\cfrac{\cfrac{\emptyset}{A \supset B \vdash A \supset B}}{A \supset B, \neg(\neg A \vee B) \vdash A \supset B} \qquad \cfrac{(3)\ \cfrac{\cfrac{\cfrac{\emptyset}{\neg A \vdash \neg A}}{\neg A \vdash \neg A \vee B}}{\neg(\neg A \vee B) \vdash A}}{A \supset B, \neg(\neg A \vee B) \vdash A}}{A \supset B, \neg(\neg A \vee B) \vdash B}$$

Examples

$$(1)\ \cfrac{\cfrac{\cfrac{\cfrac{\emptyset}{B \vdash B}}{B \vdash \neg A \vee B}}{\neg(\neg A \vee B) \vdash \neg B}}{A \supset B, \neg(\neg A \vee B) \vdash \neg B}$$

$$\cfrac{\cfrac{A \supset B, \neg(\neg A \vee B) \vdash B \qquad A \supset B, \neg(\neg A \vee B) \vdash \neg B}{A \supset B \vdash \neg\neg(\neg A \vee B)}}{A \supset B \vdash \neg A \vee B}$$

Examples

To prove (11) we have to prove the following:

$$\neg A \vee B \vdash A \supset B \tag{13}$$

$$\cfrac{\cfrac{(9)}{\neg A \vdash A \supset B} \qquad \cfrac{(10)}{B \vdash A \supset B}}{\neg A \vee B \vdash A \supset B}$$

Examples

$$A \supset B, \neg B \vdash \neg A \tag{14}$$

$$A \supset B \vdash \neg B \supset \neg A \tag{15}$$

Proofs of (14), (15):

$$\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A \supset B, A, \neg B \vdash \neg B}}{A \supset B, \neg B, A \vdash \neg B} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A, A \supset B \vdash B}}{A \supset B, A \vdash B}}{A \supset B, A, \neg B \vdash B}}{A \supset B, \neg B, A \vdash B}}{A \supset B, \neg B \vdash \neg A}}{A \supset B \vdash \neg B \supset \neg A}$$

Examples

$$\neg B \supset \neg A \vdash A \supset B \tag{16}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{\neg B \supset \neg A, \neg B, A \vdash A} \qquad \dfrac{\dfrac{\emptyset}{\neg B \supset \neg A, \neg B \vdash \neg A}}{\neg B \supset \neg A, \neg B, A \vdash \neg A}}{\neg B \supset \neg A, A \vdash \neg\neg B}}{\neg B \supset \neg A, A \vdash B}}{\neg B \supset \neg A \vdash A \supset B}$$

Examples

On the base of (15), (16):

$$\vdash A \supset B \equiv \neg B \supset \neg A \tag{17}$$

Proof:

$$\frac{A \supset B \vdash \neg B \supset \neg A \qquad \neg B \supset \neg A \vdash A \supset B}{\vdash A \supset B \equiv \neg B \supset \neg A}$$

Example

$$\vdash (A \vee \neg A) \tag{18}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\emptyset}{A, \neg(A \vee \neg A) \vdash \neg(A \vee \neg A)}}{\neg(A \vee \neg A), A \vdash \neg(A \vee \neg A)} \qquad \dfrac{\dfrac{\emptyset}{\neg(A \vee \neg A), A \vdash A}}{\neg(A \vee \neg A), A \vdash A \vee \neg A}}{\neg(A \vee \neg A) \vdash \neg A}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{\neg A, \neg(A \vee \neg A) \vdash \neg(A \vee \neg A)}}{\neg(A \vee \neg A), \neg A \vdash \neg(A \vee \neg A)} \qquad \dfrac{\dfrac{\emptyset}{\neg(A \vee \neg A), \neg A \vdash \neg A}}{\neg(A \vee \neg A), \neg A \vdash A \vee \neg A}}{\neg(A \vee \neg A) \vdash \neg\neg A}}{\neg(A \vee \neg A) \vdash A}$$

Examples

$$\frac{\dfrac{\neg(A \vee \neg A) \vdash \neg A \qquad \neg(A \vee \neg A) \vdash A}{\vdash \neg\neg(A \vee \neg A)}}{\vdash (A \vee \neg A)}$$

Examples

$$A \wedge B \vdash B \wedge A \tag{19}$$

Proof:

$$\frac{\dfrac{\dfrac{\emptyset}{A, B \vdash B} \qquad \dfrac{\dfrac{\emptyset}{B, A \vdash A}}{A, B \vdash A}}{A, B \vdash B \wedge A}}{A \wedge B \vdash B \wedge A}$$

Examples

$$A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C) \tag{20}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{B, A \vdash A}}{A, B \vdash A} \quad \dfrac{\emptyset}{A, B \vdash B}}{A, B \vdash A \wedge B}}{A, B \vdash (A \wedge B) \vee (A \wedge C)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{C, A \vdash A}}{A, C \vdash A} \quad \dfrac{\emptyset}{A, C \vdash C}}{A, C \vdash A \wedge C}}{A, C \vdash (A \wedge B) \vee (A \wedge C)}}{\dfrac{A, B \vee C \vdash (A \wedge B) \vee (A \wedge C)}{A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)}}$$

Examples

$$(A \wedge B) \vee (A \wedge C) \vdash A \wedge (B \vee C) \tag{21}$$

Proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{B, A \vdash A}}{A, B \vdash A}}{A \wedge B \vdash A} \quad \dfrac{\dfrac{\dfrac{\emptyset}{C, A \vdash A}}{A, C \vdash A}}{A \wedge C \vdash A}}{(A \wedge B) \vee (A \wedge C) \vdash A} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A, B \vdash B}}{A \wedge B \vdash B}}{A \wedge B \vdash B \vee C} \quad \dfrac{\dfrac{\dfrac{\emptyset}{A, C \vdash C}}{A \wedge C \vdash C}}{A \wedge C \vdash B \vee C}}{(A \wedge B) \vee (A \wedge C) \vdash B \vee C}}{(A \wedge B) \vee (A \wedge C) \vdash A \wedge (B \vee C)}$$

On the base of (20) and (21):

$$\vdash A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C) \tag{22}$$

Examples

$$\vdash A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C) \tag{23}$$

Proof: At first let us prove the following:

$$A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C) \tag{24}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A \vdash A}}{A \vdash A \vee B} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{B \vdash B}}{B, C \vdash B}}{B, C \vdash A \vee B}}{B \wedge C \vdash A \vee B}}{A \vee (B \wedge C) \vdash A \vee B} \qquad \dfrac{\dfrac{\dfrac{\emptyset}{A \vdash A}}{A \vdash A \vee C} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{C \vdash C}}{C \vdash A \vee C}}{B, C \vdash A \vee C}}{B \wedge C \vdash A \vee C}}{A \vee (B \wedge C) \vdash A \vee C}}{A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C)}$$

Examples

Now let us prove the following:

$$(A \vee B) \wedge (A \vee C) \vdash A \vee (B \wedge C) \tag{25}$$

$$\dfrac{\dfrac{\dfrac{\emptyset}{A \vdash A}}{A \vdash A \vee (B \wedge C)}}{A \vee B, A \vdash A \vee (B \wedge C)}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A \vdash A}}{A \vdash A \vee (B \wedge C)}}{A, C \vdash A \vee (B \wedge C)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{B \vdash B}}{B, C \vdash B} \qquad \dfrac{\dfrac{\emptyset}{C \vdash C}}{B, C \vdash C}}{B, C \vdash B \wedge C}}{B, C \vdash A \vee (B \wedge C)}}{A \vee B, C \vdash A \vee (B \wedge C)}$$

Examples

$$\dfrac{\dfrac{A \vee B, A \vdash A \vee (B \wedge C) \qquad A \vee B, C \vdash A \vee (B \wedge C)}{A \vee B, A \vee C \vdash A \vee (B \wedge C)}}{(A \vee B) \wedge (A \vee C) \vdash A \vee (B \wedge C)}$$

Examples

$$\vdash (A \supset B) \supset (B \supset C) \supset (A \supset C) \qquad (26)$$

Prove:
We can use the proved sequence (6).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \supset B, A \vdash B \qquad B, B \supset C \vdash C}{A \supset B, A, B \supset C \vdash C}}{A \supset B, B \supset C, A \vdash C}}{A \supset B, B \supset C \vdash A \supset C}}{A \supset B \vdash (B \supset C) \supset (A \supset C)}}{\vdash (A \supset B) \supset (B \supset C) \supset (A \supset C)}$$

Examples

$$\vdash (A \supset B) \supset (A \supset (B \supset C)) \supset (A \supset C) \tag{27}$$

Proof: The proved sequence (6) can be used:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A, A \supset B \vdash B}{A, A \supset B, A \supset (B \supset C) \vdash B} \qquad \dfrac{A, A \supset (B \supset C) \vdash B \supset C}{A, A \supset B, A \supset (B \supset C) \vdash B \supset C}}{A, A \supset B, A \supset (B \supset C) \vdash C}}{A \supset B, A \supset (B \supset C) \vdash A \supset C}}{A \supset B \vdash (A \supset (B \supset C)) \supset (A \supset C)}}{\vdash (A \supset B) \supset (A \supset (B \supset C)) \supset (A \supset C)}$$

Examples

De Morgan's laws:

$$\vdash \neg(A \wedge B) \equiv (\neg A \vee \neg B) \tag{28}$$

$$\vdash \neg(A \vee B) \equiv (\neg A \wedge \neg B) \tag{29}$$

Examples

To prove (28) at first we have to prove the following:

$$\neg(A \land B) \vdash (\neg A \lor \neg B) \tag{30}$$

$$(3)\ \dfrac{(3)\ \dfrac{\dfrac{\dfrac{\emptyset}{\neg A \vdash \neg A}}{\neg A \vdash \neg A \lor \neg B}}{\neg(\neg A \lor \neg B) \vdash A} \qquad (3)\ \dfrac{\dfrac{\dfrac{\emptyset}{\neg B \vdash \neg B}}{\neg B \vdash \neg A \lor \neg B}}{\neg(\neg A \lor \neg B) \vdash B}}{\dfrac{\neg(\neg A \lor \neg B) \vdash A \land B}{\neg(A \land B) \vdash \neg A \lor \neg B}}$$

Examples

To prove (28) we have to prove the following:

$$\neg A \lor \neg B \vdash \neg(A \land B) \tag{31}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{A \vdash A}}{A, B \vdash A}}{A \land B \vdash A}}{\neg A \lor \neg B, A \land B \vdash A} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{\neg A \vdash \neg A}}{B, \neg A \vdash \neg A} \qquad \dfrac{(8)}{B, \neg B \vdash \neg A}}{B, \neg A \lor \neg B \vdash \neg A}}{\neg A \lor \neg B, B \vdash \neg A}}{\neg A \lor \neg B, A, B \vdash \neg A}}{\neg A \lor \neg B, A \land B \vdash \neg A}}{\neg A \lor \neg B \vdash \neg(A \land B)}$$

Examples

To prove (29) at first we can prove the following:

$$\neg(A\vee B)\vdash\neg A\wedge\neg B \tag{32}$$

$$\dfrac{(1)\,\dfrac{\dfrac{\dfrac{\emptyset}{A\vdash A}}{A\vdash A\vee B}}{\neg(A\vee B)\vdash\neg A}\qquad (1)\,\dfrac{\dfrac{\dfrac{\emptyset}{B\vdash B}}{B\vdash A\vee B}}{\neg(A\vee B)\vdash\neg B}}{\neg(A\vee B)\vdash\neg A\wedge\neg B}$$

Examples

To prove (29) we have to prove the following:

$$\neg A\wedge\neg B\vdash\neg(A\vee B) \tag{33}$$

$$(2)\,\dfrac{\dfrac{(2)\,\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{\neg A\vdash\neg A}}{\neg A,\neg B\vdash\neg A}}{\neg A\wedge\neg B\vdash\neg A}}{A\vdash\neg(\neg A\wedge\neg B)}\qquad (2)\,\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{\neg B\vdash\neg B}}{\neg A,\neg B\vdash\neg B}}{\neg A\wedge\neg B\vdash\neg B}}{B\vdash\neg(\neg A\wedge\neg B)}}{A\vee B\vdash\neg(\neg A\wedge\neg B)}}{\neg A\wedge\neg B\vdash\neg(A\vee B)}$$

- A computer program that searches for a model for a propositional formula is called a SAT Solver.
- When you build a truth table for an unsatisfiable formula of size n you will have to generate all 2^n rows, but if the formula is satisfiable you might get lucky and find a model after generating only a few rows.
- Even an incomplete algorithm – one that can find a model if one exists but may not be able to detect if a formula is unsatisfiable – can be useful in practice.
- Many problems in computer science can be encoded in propositional logic so that a model for a formula is a solution to the problem.

Definition

- A clause is a set of literals.
- A clause is considered to be an implicit disjunction of its literals.
- A unit clause is a clause consisting of exactly one literal.
- The empty set of literals is the empty clause, denoted by $\square$.
- A formula in clausal form is a set of clauses. A formula is considered to be an implicit conjunction of its clauses.
- The formula that is the empty set of clauses is denoted by $\emptyset$.

- The only significant difference between clausal form and the standard syntax is that clausal form is defined in terms of sets, while our standard syntax was defined in terms of trees.
- A node in a tree may have multiple children that are identical subtrees, but a set has only one occurrence of each of its elements. However, this difference is of no logical significance: $(p \Leftrightarrow p \vee p)$

Theorem

- Every formula A in propositional logic can be transformed into an logically equivalent formula in clausal form.

- A clause if trivial if it contains a pair of clashing literals (positive and negative literals with the same base).
- Let S be a set of clauses and let (i.e. a clausal form), and let $C \in S$ a trivial clause Then $S \setminus \{C\}$ is logically equivalent to S (i.e. corresponding formulas are logically equivalent).

Lemma

- $\square$, the empty clause, is unsatisfiable. (The corresponding formula is unsatisfiable.)
- $\emptyset$, the empty set of clauses, is valid.

- A clause is satisfiable iff there is some interpretation under which at least one literal in the clause is true. Let ϱ be an arbitrary interpretation. Since there are no literals in $\square$, there are no literals whose value is true under ϱ. But ϱ was an arbitrary interpretation, so $\square$ is unsatisfiable
- A set of clauses is valid iff every clause in the set is true in every interpretation. But there are no clauses in $\emptyset$ that need be true, so $\emptyset$ is valid.

Definition

Let S, S' be sets of clauses. $S \approx S'$ denotes that S is satisfiable if and only if S' is satisfiable.

Remark

It is important to understand that $S \approx S'$ (S is satisfiable if and only if S' is satisfiable) does not imply that $S \Leftrightarrow S'$ (S is logically equivalent to S').

- $S = \{pq\bar{r}, p\bar{q}, \bar{p}q\}$, $S' = \{p\bar{q}, \bar{p}q\}$
- $\varrho(p) = 0, \varrho(q) = 0, \varrho(r) = 1$

Definition

- if l is a literal then l^c is its complement:
  - if $l = p$, then $l^c = \neg p$;
  - if $l = \neg p$, then $l^c = p$.
- Let S be a set of clauses. A pure literal in S is a literal l that appears in at least one clause of S, but its complement l^c does not appear in any clause of S.

TheoremI

Let S be a set of clauses and let l be a pure literal in S. Let S' be obtained from S by deleting every clause containing l. Then $S \approx S'$.

Theorem (Unit clause)

Let $\{l\} \in S$ be a unit clause and let S' be obtained from S by deleting every clause containing l and by deleting l^c from every (remaining) clause. Then $S \approx S'$.

Proof

- It is trivial from the definition of clause form.
- $S = \{r, pq\bar{r}, p\bar{q}, \bar{p}q\}$, $S' = \{pq, p\bar{q}, \bar{p}q\}$

Corollary

$\square$ is unsatisfiable.

$\{\{p\}, \{\bar{p}\}\}$ is the clausal form of the unsatisfiable formula $p \wedge \neg p$. Delete the first clause $\{p\}$ from the formula and the literal $\bar{p}$ from the second clause; the result is $\{\{\}\} = \{\}$. $\{\} \approx \{\{p\}, \{\bar{p}\}\}$ and therefore $\square$ is unsatisfiable.

Definition

Let C_1, C_2 be two clauses. If $C_1 \subseteq C_2$, then clause C_1 subsumes the clause C_2 and C_2 is subsumed by C_1.

Ttel

Let $C_1, C_2 \in S$, C_1 subsumes C_2, $S' = S \setminus \{C_2\}$. Then $S \approx S'$.

Example

$S = \{pr, \bar{p}q\bar{r}, q\bar{r}\}$, $S' = \{pr, q\bar{r}\}$

Definition

Let S be a set of clauses and U a set of atomic propositions. $R_U(S)$, the renaming of S by U, is obtained from S by replacing each literal l on an atomic proposition in U by l^c.

Theorem

$S \approx R_U(S)$

Definition

Let C_1, C_2 be clauses such that $l \in C_1, l^c \in C_2$. The clauses C_1, C_2 are said to be clashing clauses and to clash on the complementary pair of literals l, l^c.

Resolution rule

Let C_1, C_2 be clauses such that $l \in C_1, l^c \in C_2$. C, the resolvent of C_1 and C_2, is the clause: $Res(C_1, C_2) = (C_1 \setminus \{l\}) \cup (C_2 \setminus \{l^c\})$.
C_1 and C_2 are the parent clauses of C.

Example

- The pair of clauses $C_1 = ab\bar{c}$ and $C_2 = bc\bar{e}$ clash on the pair of complementary literals $c, \bar{c}$. The resolvent is:
  $C = (ab\bar{c}\{\bar{c}\}) \cup (bc\bar{e} \setminus \{c\}) = ab \cup b\bar{e} = ab\bar{e}$.
- Recall that a clause is a set so duplicate literals are removed when taking the union:
  $\{a, b\} \cup \{b, \bar{e}\} = \{a, b, \bar{e}\}$.
- Resolution is only performed if the pair of clauses clash on exactly one pair of complementary literals.

Theorem

If two clauses clash on more than one literal, their resolvent is a trivial clause.

Example

Consider a pair of clauses:

$\{l_1, l_2\} \cup C_1, \{l_1^c, l_2^c\} \cup C_2,$

and suppose that we perform the resolution rule because the clauses clash on the pair of literals $\{l_1, l_1^c\}$. The resolvent is the trivial clause: $\{l_2, l_2^c\} \cup C_1 \cup C_2$.

Theorem

The resolvent C is satisfiable if and only if the parent clauses C_1 and C_2 are both satisfiable.

Resolution algorithm

- Input: A set of clauses S.
- Output: S is satisfiable or unsatisfiable.
- Let S be a set of clauses and define $S_0 = S$.
- Repeat the following steps to obtain Si+1 from Si until the procedure terminates as defined below:
  - Choose a pair of clashing clauses $\{C_1, C_2\} \subseteq S_i$ that has not been chosen before.
  - Compute $C = Res(C_1, C_2)$ according to the resolution rule.
  - If C is not a trivial clause, let $S_{i+1} = S_i \cup \{C\}$; otherwise, $S_{i+1} = S_i$.
- Terminate the procedure if:
  - $C = \Box$
  - All pairs of clashing clauses have be resolved.

Example

Consider the set of clauses:
$S = \{(1)p, (2)\bar{p}q, (3)\bar{r}, (4)\bar{p}\bar{q}r\}$,
where the clauses have been numbered. Here is a resolution derivation of $\square$ from S, where the justification for each line is the pair of the numbers of the parent clauses that have been resolved to give the resolvent clause:

(5) $\bar{p}\bar{q}$ from $(3), (4)$
(6) $\bar{p}$ from $(5), (2)$
(7) $\square$ from $(6), (1)$

Definition

A derivation of $\square$ from a set of clauses S is a refutation by resolution of S or a resolution refutation of S.

Theorem

If the set of clauses labeling the leaves of a resolution tree is satisfiable then the clause at the root is satisfiable.

Theorem (Soundness)

Let S be a set of clauses. If there is a refutation by resolution for S then S is unsatisfiable.

Theorem (Completeness)

If a set of clauses is unsatisfiable then the empty clause $\square$ will be derived by the resolution procedure.

Davis-Putnam algorithm

- Input: A formula A in clausal form.
- Output: Report that A is satisfiable or unsatisfiable.
- Perform the following rules repeatedly, but the third rule is used only if the first two do not apply:
  - Unit-literal rule: If there is a unit clause $\{l\}$, delete all clauses containing l and delete all occurrences of l^c from all other clauses.
  - Pure-literal rule: If there is a pure literal l, delete all clauses containing l.
  - Eliminate a variable by resolution: Choose an atom p and perform all possible resolutions on clauses that clash on p and $\bar{p}$. Add these resolvents to the set of clauses and then delete all clauses containing p or $\bar{p}$.
- Terminate the algorithm under the following conditions:
  - If empty clause $\Box$ is produced, report that the formula is unsatisfiable.
  - If no more rules are applicable, report that the formula is satisfiable.

Example

- Consider the set of clauses: $\{p, pq, qr, rst\}$.
- Performing the unit-literal rule on p leads to the creation of a new unit clause q upon which the rule can be applied again.
- This leads to a new unit clause r and applying the rule results in the singleton set of clauses $\{st\}$.
- Since no more rules are applicable, the set of clauses is satisfiable.

Definition

Repeatedly applying the unit-literal rule until it is no longer applicable is called unit propagation or Boolean constraint propagation.

Definition

Let A be a set of clauses and let ϱ be a partial interpretation for A. For $C \in A$, if $|C|_\varrho = 1$, the interpretation ϱ satisfies C, while if $|C|_\varrho = 0$, then C is a conflict clause for ϱ.

DPLL algorithm

- Input: A formula A in clausal form.
- Output: Report that A is unsatisfiable or report that A is satisfiable and return a partial interpretation that satisfies A.

The algorithm is expressed as the recursive function $DPLL(B, \varrho)$ which takes two parameters: a formula B in clausal form and a partial interpretation ϱ. It is initially called with the formula A and the empty partial interpretation.

Remark

The DPLL algorithm is highly nondeterministic: it must choose an unassigned atom and then choose which truth value will be assigned to it first.

$DPLL(B, \varrho)$

- Construct the set of clauses B' by performing unit propagation on B. Construct ϱ' by adding to ϱ all the assignments made during propagation.
- Evaluate B' under the partial interpretation ϱ':
  - If B' contains a conflict clause return 'unsatisfiable';
  - If B' is satisfied return ϱ';
  - (otherwise, continue).
- Choose an atom p in B'; choose a truth value val as 1 or 0; ϱ_1 is the interpretation ϱ' together with the assignment of val to p.
- $result \leftarrow DPLL(B', \varrho_1)$
  - If $result$ is not unsatisfiable return $result$;
  - (otherwise, continue).
- ϱ_2 is the interpretation ϱ_1 together with the assignment of the complement of val to p.
- $result \leftarrow DPLL(B', \varrho_2)$
  - Return $result$.

Definition/1

The language of first–order logic is a

$$L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$$

ordered 5–tuple, where

1. $LC = \{\neg, \supset, \wedge, \vee, \equiv, =, \forall, \exists, (,)\}$: (the set of logical constants).
2. Var $(= \{x_n : n = 0, 1, 2, \ldots\})$: countable infinite set of variables

Definition/2

3. $Con = \bigcup_{n=0}^{\infty}(\mathcal{F}(n) \cup \mathcal{P}(n))$ the set of non–logical constants (at best countable infinite)
   - $\mathcal{F}(0)$: the set of name parameters,
   - $\mathcal{F}(n)$: the set of n argument function parameters,
   - $\mathcal{P}(0)$: the set of prposition parameters,
   - $\mathcal{P}(n)$: the set of predicate parameters.
4. The sets LC, Var, $\mathcal{F}(n)$, $\mathcal{P}(n)$ are pairwise disjoint $(n = 0, 1, 2, \dots)$.

Definition/3

5. The set of terms, i.e. the set *Term* is given by the following inductive definition:
   (a) $Var \cup \mathcal{F}(0) \subseteq Term$
   (b) If $f \in \mathcal{F}(n)$, $(n = 1, 2, \dots)$, s $t_1, t_2, \dots, t_n \in Term$, then $f(t_1, t_2, \dots, t_n) \in Term$.

Definition/4

6. The set of formulas, i.e. the set *Form* is given by the following inductive definition:
   (a) $\mathcal{P}(0) \subseteq Form$
   (b) If $t_1, t_2 \in Term$, then $(t_1 = t_2) \in Form$
   (c) If $P \in \mathcal{P}(n)$, $(n = 1, 2, \dots)$, s $t_1, t_2, \dots, t_n \in Term$, then $P(t_1, t_2, \dots, t_n) \in Form$.
   (d) If $A \in Form$, then $\neg A \in Form$.
   (e) If $A, B \in Form$, then $(A \supset B)$, $(A \wedge B)$, $(A \vee B)$, $(A \equiv B) \in Form$.
   (f) If $x \in Var$, $A \in Form$, then $\forall x A$, $\exists x A \in Form$.

Definition (interpretation)

The ordered pair $\langle U, \varrho \rangle$ is an interpretation of the language $L^{(1)}$ if

- $U \neq \emptyset$ (i.e. U is a nonempty set);
- $Dom(\varrho) = Con$
  - If $a \in \mathcal{F}(0)$, then $\varrho(a) \in U$;
  - If $f \in \mathcal{F}(n)$ $(n \neq 0)$, then $\varrho(f) \in U^{U^{(n)}}$
  - If $p \in \mathcal{P}(0)$, then $\varrho(p) \in \{0, 1\}$;
  - If $P \in \mathcal{P}(n)$ $(n \neq 0)$, then $\varrho(P) \subseteq U^{(n)}$ $(\varrho(P) \in \{0, 1\}^{U^{(n)}})$.

Definition (assignment)

The function v is an assignment relying on the interpretation $\langle U, \varrho \rangle$ if the followings hold:

- $Dom(v) = Var$;
- If $x \in Var$, then $v(x) \in U$.

Definition (modified assignment)

Let v be an assignment relying on the interpretation $\langle U, \varrho \rangle$, $x \in Var$ and $u \in U$.

$$v[x : u](y) = \begin{cases} u, & \text{if } y = x; \\ v(y), & \text{otherwise.} \end{cases}$$

for all $y \in Var$.

Definition (Semantic rules/1)

Let $\langle U, \varrho \rangle$ be a given interpretation and v be an assignment relying on $\langle U, \varrho \rangle$.

- If $a \in \mathcal{F}(0)$, then $|a|_v^{\langle U, \varrho \rangle} = \varrho(a)$.
- If $x \in Var$, then $|x|_v^{\langle U, \varrho \rangle} = v(x)$.
- If $f \in \mathcal{F}(n)$, $(n = 1, 2, \dots)$, and $t_1, t_2, \dots, t_n \in Term$, then $|f(t_1)(t_2)\dots(t_n)|_v^{\langle U, \varrho \rangle} = \varrho(f)(\langle |t_1|_v^{\langle U, \varrho \rangle}, |t_2|_v^{\langle U, \varrho \rangle}, \dots, |t_n|_v^{\langle U, \varrho \rangle} \rangle)$
- If $p \in \mathcal{P}(0)$, then $|p|_v^{\langle U, \varrho \rangle} = \varrho(p)$
- If $t_1, t_2 \in Term$, then

$$|(t_1 = t_2)|_v^{\langle U, \varrho \rangle} = \begin{cases} 1, & \text{if } |t_1|_v^{\langle U, \varrho \rangle} = |t_2|_v^{\langle U, \varrho \rangle} \\ 0, & \text{otherwise.} \end{cases}$$

Definition (Semantic rules/2)

- If $P \in \mathcal{P}(n)$ $(n \neq 0)$, $t_1, \ldots, t_n \in Term$, then

$$|P(t_1)\ldots(t_n)|_v^{\langle U,\varrho\rangle} = \begin{cases} 1, & \text{if } \langle |t_1|_v^{\langle U,\varrho\rangle}, \ldots, |t_n|_v^{\langle U,\varrho\rangle} \rangle \in \varrho(P); \\ 0, & \text{otherwise.} \end{cases}$$

Definition (Semantic rules/3)

- If $A \in Form$, then $|\neg A|_v^{\langle U,\varrho\rangle} = 1 - |A|_v^{\langle U,\varrho\rangle}$.
- If $A, B \in Form$, then

$$|(A \supset B)|_v^{\langle U,\varrho\rangle} = \begin{cases} 0 & \text{if } |A|_v^{\langle U,\varrho\rangle} = 1, \text{ and } |B|_v^{\langle U,\varrho\rangle} = 0; \\ 1, & \text{otherwise.} \end{cases}$$

$$|(A \wedge B)|_v^{\langle U,\varrho\rangle} = \begin{cases} 1 & \text{if } |A|_v^{\langle U,\varrho\rangle} = 1, \text{ and } |B|_v^{\langle U,\varrho\rangle} = 1; \\ 0, & \text{otherwise.} \end{cases}$$

$$|(A \vee B)|_v^{\langle U,\varrho\rangle} = \begin{cases} 0 & \text{if } |A|_v^{\langle U,\varrho\rangle} = 0, \text{ and } |B|_v^{\langle U,\varrho\rangle} = 0; \\ 1, & \text{otherwise.} \end{cases}$$

$$|(A \equiv B)|_v^{\langle U,\varrho\rangle} = \begin{cases} 1 & \text{if } |A|_v^{\langle U,\varrho\rangle} = |B|_v^{\langle U,\varrho\rangle} = 0; \\ 0, & \text{otherwise.} \end{cases}$$

Definition (Semantic rules/4)

- If $A \in Form, x \in Var$, then

$$|\forall x A|_v^{\langle U,\varrho\rangle} = \begin{cases} 0, & \text{if there is an } u \in U \text{ such that } |A|_{v[x:u]}^{\langle U,\varrho\rangle} = 0; \\ 1, & \text{otherwise.} \end{cases}$$

$$|\exists x A|_v^{\langle U,\varrho\rangle} = \begin{cases} 1, & \text{if there is an } u \in U \text{ such that } |A|_{v[x:u]}^{\langle U,\varrho\rangle} = 1; \\ 0, & \text{otherwise.} \end{cases}$$

Definition (model – a set of formulas)

Let $L(1) = \langle LC, Var, Con, Term, Form\rangle$ be a first order language and $\Gamma \subseteq Form$ be a set of formulas. An ordered triple $\langle U, \varrho, v\rangle$ is a model of the set Γ, if

- $\langle U, \varrho\rangle$ is an interpretation of $L^{(1)}$;
- v is an assignment relying on $\langle U, \varrho\rangle$;
- $|A|_v^{\langle U,\varrho\rangle} = 1$ for all $A \in \Gamma$.

Definition – a model of a formula

A model of a formula A is the model of the singleton $\{A\}$.

Definition – satisfiable a set of formulas

The set of formulas $\Gamma \subseteq Form$ is satisfiable if it has a model.
(If there is an interpretation and an assignment in which all members of the set Γ are true.)

Definition – satisfiable a formula

A formula $A \in Form$ is satisfiable, if the singleton $\{A\}$ is satisfiable.

Remark

- A satisfiable set of formulas does not involve a logical contradiction; its formulas may be true together.
- A satisfiable formula may be true.
- If a set of formulas is satisfiable, then its members are satisfiable.
- But: all members of the set $\{P(a), \neg P(a)\}$ are satisfiable, and the set is not satisfiable.

Theorem

All subsets of a satisfiable set are satisfiable.

Proof

- Let $\Gamma \subseteq Form$ be a set of formulas and $\Delta \subseteq \Gamma$.
- Γ is satisfiable: it has a model. Let $\langle U, \varrho, v\rangle$ be a model of Γ.
- A property of $\langle U, \varrho, v\rangle$: If $A \in \Gamma$, then $|A|_v^{\langle U,\varrho\rangle} = 1$
- Since $\Delta \subseteq \Gamma$, if $A \in \Delta$, then $A \in \Gamma$, and so $|A|_v^{\langle U,\varrho\rangle} = 1$. That is the ordered triple $\langle U, \varrho, v\rangle$ is a model of Δ, and so Δ is satisfiable.

Definition – unsatisfiable set

The set $\Gamma \subseteq Form$ is unsatisfiable if it is not satisfiable.

Definition – unsatisfiable formula

A formula $A \in Form$ is unsatisfiable if the singleton $\{A\}$ is unsatisfiable.

Remark

A unsatisfiable set of formulas involve a logical contradiction. (Its members cannot be true together.)

Theorem

All expansions of an unsatisfiable set of formulas are unsatisfiable.

Indirect proof

- Suppose that $\Gamma \subseteq Form$ is an unsatisfiable set of formulas and $\Delta \subseteq Form$ is a set of formulas.
- Indirect condition: Γ is unsatisfiable, and $\Gamma \cup \Delta$ satisfiable.
- $\Gamma \subseteq \Gamma \cup \Delta$
- According to the former theorem Γ is satisfiable, and it is a contradiction.

Definition

A formula A is the logical consequence of the set of formulas Γ if the set $\Gamma \cup \{\neg A\}$ is unsatifiable. $(Notation : \Gamma \vDash A)$

Definition

$A \vDash B$, if $\{A\} \vDash B$.

Definition

The formula A is valid if $\emptyset \vDash A$. (Notation: $\vDash A$)

Definition

The formulas A and B are logically equivalent if $A \vDash B$ and $B \vDash A$. (Notation: $A \Leftrightarrow B$)

Theorem

Let $\Gamma \subseteq \mathit{Form}$, and $A \in \mathit{Form}$. $\Gamma \vDash A$ if and only if all models of the set Γ are the models of formula A. (i.e. the singleton $\{A\}$).

Proof

$\rightarrow$ Indirect condition: There is a model of $\Gamma \vDash A$ such that it is not a model of the formula A.
Let the ordered triple $\langle U, \varrho, v\rangle$ be this model.
The properties of $\langle U, \varrho, v\rangle$:

1. $|B|_v^{\langle U,\varrho\rangle} = 1$ for all $B \in \Gamma$;
2. $|A|\langle U, \varrho\rangle_v = 0$, and so $|\neg A|_v^{\langle U,\varrho\rangle} = 1$

In this case all members of the set $\Gamma \cup \{\neg A\}$ are true wrt the interpretation $\langle U, \varrho\rangle$ and assignment v, so $\Gamma \cup \{\neg A\}$ is satisfiable. It means that $\Gamma \nvDash A$, and it is a contradiction.

Proof

$\leftarrow$ Indirect condition: All models of the set Γ are the models of formula A, but (and) $\Gamma \nvDash A$.
In this case $\Gamma \cup \{\neg A\}$ is satisfiable, i.e. it has a model.
Let the ordered triple $\langle U, \varrho, v\rangle$ be a model.
The properties of $\langle U, \varrho, v\rangle$:

1. $|B|_v^{\langle U,\varrho\rangle} = 1$ for all $B \in \Gamma$;
2. $|\neg A|_v^{\langle U,\varrho\rangle} = 1$, i.e. $|A|_v^{\langle U,\varrho\rangle} = 0$

So the set Γ has a model such that it is not a model of formula A, and it is a contradiction.

Corollary

Let $\Gamma \subseteq \mathit{Form}$, and $A \in \mathit{Form}$. $\Gamma \vDash A$ if and only if for all interpretations in which all members of Γ are true, the formula A is true.

Theorem

If A is a valid formula $((\vDash A))$, then $\Gamma \vDash A$ for all sets of formulas Γ. (A valid formula is a consequence of any set of formulas.)

Proof

- If A is a valid formula, then $\emptyset \vDash A$ (according to its definition).
- $\emptyset \cup \{\neg A\}$ $(= \{\neg A\})$ is unsatisfiable, and so its expansions are unsatisfiable.
- $\Gamma \cup \{\neg A\}$ is an expansion of $\{\neg A\}$, and so it is unsatisfiable, i.e. $\Gamma \vDash A$.

Theorem

If Γ is unsatisfiable, then $\Gamma \vDash A$ for all A. (All formulas are the consequences of an unsatisfiable set of formulas.)

Proof

- According to a proved theorem: If Γ is unsatisfiable, the all expansions of Γ are unsatisfiable.
- $\Gamma \cup \{\neg A\}$ is an expansion of Γ, and so it is unsatisfiable, i.e. $\Gamma \vDash A$.

Theorem

Deduction theorem: If $\Gamma \cup \{A\} \vDash B$, then $\Gamma \vDash (A \supset B)$.

Proof

- Indirect condition: Suppose, that $\Gamma \cup \{A\} \vDash B$, and $\Gamma \nvDash (A \supset B)$.
- $\Gamma \cup \{\neg(A \supset B)\}$ is satisfiable, and so it has a model. Let the ordered triple $\langle U, \varrho, v\rangle$ be a model.
- The properties of $\langle U, \varrho, v\rangle$:
  1. All members of Γ are true wrt $\langle U, \varrho\rangle$ and v.
  2. $|\neg(A \supset B)|_v^{\langle U,\varrho\rangle} = 1$
- $|(A \supset B)|_v^{\langle U,\varrho\rangle} = 0$, i.e. $|A|_v^{\langle U,\varrho\rangle} = 1$ and $|B|_v^{\langle U,\varrho\rangle} = 0$. So$|\neg B|_v^{\langle U,\varrho\rangle} = 1$.
- All members of $\Gamma \cup \{A\} \cup \{\neg B\}$ are true wrt $\langle U, \varrho\rangle$ and v, i.e. $\Gamma \cup \{A\} \nvDash B$, and it is a contradiction.

Theorem

In the opposite direction: If $\Gamma \vDash (A \supset B)$, then $\Gamma \cup \{A\} \vDash B$.

Proof

- Indirect condition: Suppose that $\Gamma \vDash (A \supset B)$, and $\Gamma \cup \{A\} \nvDash B$.
- So $\Gamma \cup \{A\} \cup \{\neg B\}$ is satisfiable, i.e. it has a model. Let the ordered triple $\langle U, \varrho, v\rangle$ a model.
- The properties of $\langle U, \varrho, v\rangle$:
  1. All members of Γ are true wrt $\langle U, \varrho\rangle$ and v.
  2. $|A|_v^{\langle U,\varrho\rangle} = 1$
  3. $|\neg B|_v^{\langle U,\varrho\rangle} = 1$, and so $|B|_v^{\langle U,\varrho\rangle} = 0$
- $|(A \supset B)|_v^{\langle U,\varrho\rangle} = 0$, $|\neg(A \supset B)|_v^{\langle U,\varrho\rangle} = 1$.
- All members of $\Gamma \cup \{\neg(A \supset B)\}$ are true wrt $\langle U, \varrho\rangle$ and v, i.e. $\Gamma \nvDash (A \supset B)$.

Corollary

$A \vDash B$ if and only if $\vDash (A \supset B)$

Proof

Let $\Gamma = \emptyset$ in the former theorems.

Cut elimination theorem

If $\Gamma \cup \{A\} \vDash B$ and $\Delta \vDash A$, then $\Gamma \cup \Delta \vDash B$.

Proof

Indirect.

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula. The set of free variables of the formula A (in notation: $FreeVar(A)$) is given by the following inductive definition:

- If A is an atomic formula (i.e. $A \in AtForm$), then the members of the set $FreeVar(A)$ are the variables occuring in A.
- If the formula A is $\neg B$, then $FreeVar(A) = FreeVar(B)$.
- If the formula A is $(B \supset C)$, $(B \wedge C)$, $(B \vee C)$ or $(B \equiv C)$, then $FreeVar(A) = FreeVar(B) \bigcup FreeVar(C)$.
- If the formula A is $\forall xB$ or $\exists xB$, then $FreeVar(A) = FreeVar(B) \setminus \{x\}$.

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula. The set of bound variables of the formula A (in notation: $BoundVar(A)$) is given by the following inductive definition:

- If A is an atomic formula (i.e. $A \in AtForm$), then $BoundVar(A) = \emptyset$.
- If the formula A is $\neg B$, then $BoundVar(A) = FreeVar(B)$.
- If the formula A is $(B \supset C)$, $(B \wedge C)$, $(B \vee C)$ or $(B \equiv C)$, then $BoundVar(A) = BoundVar(B) \bigcup BoundVar(C)$.
- If the formula A is $\forall xB$ or $\exists xB$, then $BoundVar(A) = BoundVar(B) \cup \{x\}$.

Remark

- The bases of inductive definitions of sest of free and bound variables are given by the first requirement of the corresponding definitions.
- The sets of free and bound variables of a formula are not disoint necessarily:
  $FreeVar((P(x) \wedge \exists x R(x))) = \{x\} = BoundVar((P(x) \wedge \exists x R(x)))$

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A \in Form$ be a formula, and $x \in Var$ be a variable.

- A fixed occurence of the variable x in the formula A is free if it is not in the subformulas $\forall x B$ or $\exists x B$ of the formula A.
- A fixed occurence of the variable x in the formula A is bound if it is not free.

Remark

- If x is a free variable of the formula A (i.e. $x \in \mathit{FreeVar}(A)$), then it has at least one free occurence in A.
- If x is a bound variable of the formula A (i.e. $x \in \mathit{BoundVar}(A)$), then it has at least one bound occurence in A.
- A fixed occurence of a variable x in the formula A is free if
  - it does not follow a universal or an existential quantifier, or
  - it is not in a scope of a $\forall x$ or a $\exists x$ quantification.
- A variable x may be a free and a bound variable of the formula A: $(P(x) \wedge \exists x R(x))$

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order languuage and $A \in Form$ be a formula.

- If $\mathit{FreeVar}(A) \neq \emptyset$, then the formula A is an open formula.
- If $\mathit{FreeVar}(A) = \emptyset$, then the formula A is a closed formula.

Remark:
The formula A is open if there is at least one variable which has at least one free occurence in A.
The formula A is closed if there is no variable which has a free occurence in A.

De Morgan Laws of quantifications

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language, $A \in Form$ be a formula and $x \in Var$ be a variable. Then

- $\neg \exists x A \Leftrightarrow \forall x \neg A$
- $\neg \forall x A \Leftrightarrow \exists x \neg A$

Expressibilty of quantifications

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language, $A \in Form$ be a formula and $x \in Var$ be a variable. Then

- $\exists x A \Leftrightarrow \neg \forall x \neg A$
- $\forall x A \Leftrightarrow \neg \exists x \neg A$

Conjunction and quantifications

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language, $A, B \in Form$ be formulas and $x \in Var$ be a variable.
If $x \notin FreeVar(A)$, then

- $A \wedge \forall x B \Leftrightarrow \forall x(A \wedge B)$
- $A \wedge \exists x B \Leftrightarrow \exists x(A \wedge B)$

Remark:
According to the commutativity of conjunction the followings hold:
If $x \notin FreeVar(A)$, then

- $\forall x B \wedge A \Leftrightarrow \forall x(B \wedge A)$
- $\exists x B \wedge A \Leftrightarrow \exists x(B \wedge A)$

Disjunction and quantifications

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language, $A, B \in Form$ be formulas and $x \in Var$ be a variable.
If $x \notin FreeVar(A)$, then

- $A \vee \forall x B \Leftrightarrow \forall x(A \vee B)$
- $A \vee \exists x B \Leftrightarrow \exists x(A \vee B)$

Remark:
According to the commutativity of disjunction the followings hold:
If $x \notin FreeVar(A)$, then

- $\forall x B \vee A \Leftrightarrow \forall x(B \vee A)$
- $\exists x B \vee A \Leftrightarrow \exists x(B \vee A)$

Implication with existential quantification

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A, B \in Form$ be formulas and $x \in Var$ be a variable.
If $x \notin FreeVar(A)$, then

- $A \supset \exists xB \Leftrightarrow \exists x(A \vee B)$
- $\exists xB \supset A \Leftrightarrow \forall x(B \supset A)$

Implication with universal quantification

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A, B \in Form$ be formulas and $x \in Var$ be a variable.
If $x \notin FreeVar(A)$, then

- $A \supset \forall xB \Leftrightarrow \forall x(A \vee B)$
- $\forall xB \supset A \Leftrightarrow \exists x(B \supset A)$

Substitutabily a variable with an other variable

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A \in Form$ be a formula and $x, y \in Var$ be variables.
The variable x is subtitutable with the variable y in the formula A if there is no a free occurence of x in A which is in the subformulas $\forall yB$ or $\exists yB$ of A.

Example:

- In the formula $\forall zP(x, z)$ the variable x is substitutable with the variable y, but x is not substitutable with the variable z.

Substitutabily a variable with a term

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A \in Form$ be a formula, $x \in Var$ be a variable and $t \in Term$ be a term.
The variable x is subtitutable with the term t in the formula A if in the formula A the variable x is substitutable with all variables occuring in the term t.

Example

- In the formula $\forall zP(x, z)$ the variable x is substitutable with the term $f(y_1, y_2)$, but x is not substitutable with the term $f(y, z)$.

Result of a substitution

If the variable x is subtitutable with the term t in the formula A, then $[A]_x^t$ denotes the formula which appear when all free occurences of the variable x in A are substituted with the term t.

Renaming

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language, $A \in Form$ be a formula, and $x, y \in Var$ be variables.
If the variable x is subtitutable with the variable y in the formula A and $y \notin FreeVar(A)$, then

- the formula $\forall y[A]_x^y$ is a regular renaming of the formula $\forall xA$;
- the formula $\exists y[A]_x^y$ is a regular renaming of the formula $\exists xA$.

Congruent formulas

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula.
The set $Cong(A)$ (the set ot formulas which are congruent with A) is given by the following inductive definition:

- $A \in Cong(A)$;
- if $\neg B \in Cong(A)$ and $B' \in Cong(B)$, then $\neg B' \in Cong(A)$;
- if $(B \circ C) \in Cong(A)$, $B' \in Cong(B)$ and $C' \in Cong(C)$, then $(B' \circ C') \in Cong(A)$ $(\circ \in \{\supset, \wedge, \vee, \equiv\})$;
- if $\forall x B \in Cong(A)$ and $\forall y [B]_x^y$ is a regular renaming of the formula $\forall x B$, then $\forall y [B]_x^y \in Cong(A)$;
- if $\exists x B \in Cong(A)$ and $\exists y [B]_x^y$ is a regular renaming of the formula $\exists x B$, then $\exists y [B]_x^y \in Cong(A)$.

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A, B \in Form$ be formulas.

- If $B \in Cong(A)$, then the formula A is congruent with the formula B.
- If $B \in Cong(A)$, then the formula B is a syntactical synonym of the formula A.

Theorem

Congruent formulas are logically equivalent, i.e. if $B \in Cong(A)$, then $A \Leftrightarrow B$.

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula.
The formula A is standardized if

- $FreeVar(A) \bigcap BoundVar(A) = \emptyset$;
- all bound variables of the formula A have exactly one occurences next a quantifier.

Theorem

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula.
Then there is a formula $B \in Form$ such that

- the formula B is standardized;
- the formula B is congruent with the formula A, i.e. $B \in Cong(A)$.

Definition

Let $L^{(1)} = \langle LC, Var, Con, Term, Form \rangle$ be a first order language and $A \in Form$ be a formula.
The formula A is prenex if

- there is no quantifier in A or
- the formula A is in the form $Q_1x_1Q_2x_2 \ldots Q_nx_nB$ $(n = 1, 2, \ldots)$, where
  - there is no quantifier in the formula $B \in Form$;
  - $x_1, x_2 \ldots x_n \in Var$ are diffrent variables;
  - $Q_1, Q_2, \ldots, Q_n \in \{\forall, \exists\}$ are quantifiers.

Theorem

Let $L^{(1)} = \langle LC, Var, Con, Term, Form\rangle$ be a first order language and $A \in Form$ be a formula.
Then there is a formula $B \in Form$ such that

- the formula B is prenex;
- $A \Leftrightarrow B$.

Recall

- A formula of propositional logic is in conjunctive normal form (CNF) iff it is a conjunction of disjunctions of literals.
- A notational variant of CNF is clausal form:
  - the formula is represented as a set of clauses, where each clause is a set of literals.
- We now proceed to generalize CNF to first-order logic by defining a normal form that takes the quantifiers into account.

Definition

A formula is in prenex conjunctive normal form (PCNF) iff it is of the form:

$$Q_1x_1 \dots Q_nx_nA$$

where the Q_i are quantifiers and A is a quantifier-free formula in CNF. The sequence $Q_1x_1Q_nx_n$ is the prefix and A is the matrix.

Example

The following formula is in PCNF:

- $\forall y \forall z((P(f(y))\neg P(g(z)) \vee R(z)] \wedge (\neg R(z) \vee \neg P(g(z)) \vee R(y)))$
- Prefix: $\forall y \forall z$
- Matrix: $((P(f(y))\neg P(g(z)) \vee R(z)] \wedge (\neg R(z) \vee \neg P(g(z)) \vee R(y)))$

PCNF algorithm for closed formulas

- Input: A closed formula A of first-order logic.
- Output: A formula A' in PCNF such that $A' \Leftrightarrow A$.
- Steps:
  - Eliminate all binary logical operators other than $\vee$ and $\wedge$.
  - Rename bound variables so that no variable appears in two quantifiers.
  - Push negation operators inward, collapsing double negation, until they apply to atomic formulas only. Use De Morgan's laws.
  - Extract quantifiers. Choose an outermost quantifier, that is not within the scope of another quantifier. Extract the quantifier using the following equivalences, where Q is a quantifier and $\circ \in \{\vee, \wedge\}$: $A \circ QxB(x) \Leftrightarrow Qx(A \circ B(x))$, $QxA(x) \circ B \Leftrightarrow Qx(A(x) \circ B)$.
  - Use the distributive laws to transform the matrix into CNF. The formula is now in PCNF.

Definition

Let A be a closed formula in PCNF whose prefix consists only of universal quantifiers. The clausal form of A consists of the matrix of A written as a set of clauses.

Example

The formula in previous example is closed and has only universal quantifiers, so it can be written in clausal form as:

$$\{\{P(f(y)), \neg P(g(z)), R(z)\}, \{\neg R(z), \neg P(g(z)), R(y)\}\}$$

In propositional logic, every formula is equivalent to one in CNF, but this is not true in first-order logic. However, a formula in first-order logic can be transformed into one in clausal form without modifying its satisfiability.

Skolem's theorem

Let A be a closed formula. Then there exists a formula A'' in clausal form such that $A \approx A''$.

Skolem's algorithm

- Input: A closed formula A of first-order logic.
- Output: A formula A'' in clausal form such that $A'' \approx A$.
- Steps:
  - PCNF algorithm for formula A. Output is the formula A'.
  - For every existential quantifier $\exists x$ in A', let $y_1, \ldots, y_n$ be the universally quantified variables preceding $\exists x$ and let f be a new n-ary function symbol. Delete $\exists x$ and replace every occurrence of x by $f(y_1, \ldots, y_n)$. If there are no universal quantifiers preceding $\exists x$, replace x by a new constant (name parameter, 0-ary function). These new function symbols are Skolem functions and the process of replacing existential quantifiers by functions is Skolemization.
  - The formula can be written in clausal form by dropping the (universal) quantifiers and writing the matrix as sets of clauses.

Example

- Original formula: $\exists x \forall y P(x, y) \supset \forall y \exists x P(x, y)$
- Rename bound variables: $\exists x_1 \forall y_1 P(x_1, y_1) \supset \forall y_2 \exists x_2 P(x_2, y_2)$
- Eliminate Boolean operators: $\neg \exists x_1 \forall y_1 P(x_1, y_1) \vee \forall y_2 \exists x_2 P(x_2, y_2)$
- Push negation inwards: $\forall x_1 \exists y_1 \neg P(x_1, y_1) \vee \forall y_2 \exists x_2 P(x_2, y_2)$
- Extract quantifiers: $\forall x_1 \exists y_1 \forall y_2 \exists x_2 (\neg P(x_1, y_1) \vee P(x_2, y_2))$
- Distribute matrix (no change)
- Replace existential quantifiers: $\forall x_1 \forall y_2 (\neg P(x_1, f(x_1)) \vee P(g(x_1, y_2), y_2))$
- Write in clausal form $\{\{\neg P(x_1, f(x_1)), P(g(x_1, y_2), y_2)\}\}$.
- f is unary because $\exists y_1$ is preceded by one universal quantifier $\forall x_1$, while g is binary because $\exists x_2$ is preceded by two universal quantifiers $\forall x_1$ and $\forall y_2$.