



Artificial Intelligence Notes

The Foundations: Logic & Proofs

Rules of logic give precise meaning to mathematical statements. These rules are used to distinguish between valid & invalid mathematical arguments.

Proposition

Basic building block of logic. A proposition is a declarative sentence (a sentence that declares a fact) — that is either true or false, but not both.

propositional logic \Rightarrow use letters, e.g. - P, q, r, s

\vdash truth val. $\Rightarrow T$
 \dashv false val. $\Rightarrow F$

Propositional Calculus The area of logic that deals with propositions or logic

Compound proposition New propositions that are formed from existing propositions using logical operators.

Examples of compound propositions

01. The negation of $p \Rightarrow \neg p / \bar{p}$ (use negation operator)
02. Conjunction of $p \& q \Rightarrow p \wedge q$ (use connective operator)
03. Disjunction of $p \& q \Rightarrow p \vee q$ ($FF \rightarrow F, T$) [inclusive or]
04. Exclusive or $\Rightarrow p \oplus q$ ($TF / FT \rightarrow T, F$)
05. Conditional statement $\Rightarrow p \rightarrow q$ ($TF \rightarrow F, T$) [implication]
Here, $p \Rightarrow$ hypothesis / antecedent / premise
 $q \Rightarrow$ conclusion / consequence

“if p , then q ”	“ p implies q ”
“if p, q ”	“ p only if q ”
“ p is sufficient for q ”	“a sufficient condition for q is p ”
“ q if p ”	“ q whenever p ”
“ q when p ”	“ q is necessary for p ”
“a necessary condition for p is q ”	“ q follows from p ”
“ q unless $\neg p$ ”	

- *Converse : $q \rightarrow p$
- *Contrapositive : $\neg q \rightarrow \neg p$
- *Inverse : $\neg p \rightarrow \neg q$

Conditional statement \equiv Contrapositive
Converse \equiv Inverse

Bi-conditional Statement

bi-implication $p \leftrightarrow q$ ($TT / FF, T$)
* p if and only if q ; $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Operator	Precedence
\neg	1
\wedge	2
\vee	3
\rightarrow	4
\leftrightarrow	5

Precedence of Logical operators

TABLE 9 Table for the Bit Operators OR, AND, and XOR.				
x	y	$x \vee y$	$x \wedge y$	$x \oplus y$
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

*bitwise operation

Propositional Equivalences

Tautology: A compound proposition that is always true. [$p \vee \neg p$]

Contradiction: A compound proposition that is always false. [$p \wedge \neg p$]

Contingency: A compound proposition that is neither a tautology nor a contradiction.

Logically equivalent: Compound propositions, $p \equiv q$, if $p \leftrightarrow q$ is a tautology
 $\hookrightarrow p \leftrightarrow q$

TABLE 6 Logical Equivalences.

Equivalence	Name
$p \wedge T \equiv p$	
$p \vee F \equiv p$	
$p \vee T \equiv T$	Domination laws
$p \wedge F \equiv F$	
$p \vee p \equiv p$	Idempotent laws
$p \wedge p \equiv p$	
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$	Commutative laws
$p \wedge q \equiv q \wedge p$	
$(p \vee q) \vee r \equiv p \vee (q \vee r)$	Associative laws
$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	Distributive laws
$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	
$\neg(p \wedge q) \equiv \neg p \vee \neg q$	De Morgan's laws
$\neg(p \vee q) \equiv \neg p \wedge \neg q$	
$p \vee (p \wedge q) \equiv p$	Absorption laws
$p \wedge (p \vee q) \equiv p$	
$p \vee \neg p \equiv T$	Negation laws
$p \wedge \neg p \equiv F$	

TABLE 7 Logical Equivalences Involving Conditional Statements.

$p \rightarrow q \equiv \neg p \vee q$
$p \rightarrow q \equiv \neg q \rightarrow \neg p$
$p \vee q \equiv \neg p \rightarrow q$
$p \wedge q \equiv \neg(p \rightarrow \neg q)$
$\neg(p \rightarrow q) \equiv p \wedge \neg q$
$(p \rightarrow q) \wedge (p \rightarrow r) \equiv p \rightarrow (q \wedge r)$
$(p \rightarrow r) \wedge (q \rightarrow r) \equiv (p \vee q) \rightarrow r$
$(p \rightarrow q) \vee (p \rightarrow r) \equiv p \rightarrow (q \vee r)$
$(p \rightarrow r) \vee (q \rightarrow r) \equiv (p \wedge q) \rightarrow r$

TABLE 8 Logical Equivalences Involving Biconditional Statements.

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$
$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$
$p \leftrightarrow q \equiv (p \wedge q) \vee (\neg p \wedge \neg q)$
$\neg(p \leftrightarrow q) \equiv p \leftrightarrow \neg q$

* De Morgan's Laws extend to \Rightarrow

$$\neg(p_1 \vee p_2 \vee \dots \vee p_n) \equiv (\neg p_1 \wedge \neg p_2 \wedge \dots \wedge \neg p_n)$$

$$\neg(p_1 \wedge p_2 \wedge \dots \wedge p_n) \equiv (\neg p_1 \vee \neg p_2 \vee \dots \vee \neg p_n)$$

* **satisfiable:** a compound proposition is satisfiable if there is an assignment of truth val to its var. that makes it true.

Unsatisfiable: when compound proposition is false for all assignments of truth val to its var., the proposition is unsatisfiable.

Solution: solution is a particular assignment of truth val that makes compound proposition true.

Predicates & Quantifiers

Predicates: statement " x is greater than y " \Rightarrow denotes by $P(x)$

Subject Predicate - refers to a property that sub
of the statement of statement can have

propositional func
 P at x

- * once a val has been assigned to var. x , statement $P(x)$ becomes a proposition and has a truth val.
- * statement of the form $P(x_1, x_2, \dots, x_n)$ is the val of propositional function P at n-tuple (x_1, x_2, \dots, x_n)
 P is called an n-place predicates or n-ary predicate

if $x \geq 0$, then $x! = x + 1$

Pre condition Post condition

Quantifiers:

quantification: propositional function \rightarrow proposition.

it expresses the extent to which a predicate is true over a range of elements. associated english word: all, some, many,

universal quantification: it will tell that a predicate is true for every element under consideration.

existential quantification: it will tell that there is one or more element under consideration for which predicate is true.

predicate calculus: area of logic that deals \Rightarrow predicates + quantifiers.

* Universal quantified: domain of discourse / universe of discourse / domain

$\forall x P(x) \Rightarrow P(x)$ for all values of x in the domain

universal quantifier * an element for which $P(x)$ is false is called a counterexample of $\forall x P(x)$

* Existential quantifier:

existential quantifier $\exists x P(x) \Rightarrow$ there exists an element x in the domain such that $P(x)$
/ there is an x such that $P(x)$ / for some $x P(x)$
/ there is at least one x such that $P(x)$

- * a domain must always be specified when a statement $\exists x P(x)$ used. The meaning of $\exists x P(x)$ changes when domain changes.
- * $\exists x P(x)$ is false iff $P(x)$ is false for every elements of domain.

* elements of domain $\Rightarrow x_1, x_2, x_3, \dots, x_n$

universal quantification $\forall x P(x) \equiv \underbrace{P(x_1) \wedge P(x_2) \wedge P(x_3) \wedge \dots \wedge P(x_n)}_{\text{conjunction}}$

existential quantification $\exists x P(x) \equiv \underbrace{P(x_1) \vee P(x_2) \vee P(x_3) \vee \dots \vee P(x_n)}_{\text{disjunction}}$

* Uniqueness quantifier:

$\exists ! x P(x)$ or $\exists_1 x P(x) \Rightarrow$ there is one and only one x , $P(x)$

* Quantified with restricted domain:

$\forall x < 0 (x^2 > 0) \equiv \forall x (x < 0 \rightarrow x^2 > 0) \rightarrow$ conditional statement

$\exists x > 0 (x^2 = 2) \equiv \exists x (x > 0 \wedge x^2 = 2) \rightarrow$ conjunction

* Precedence of quantifiers: \forall & \exists have higher precedence than all logical operators from propositional calculus.

* Binding Var.: $\exists x (x+y=1) \Rightarrow$ here x is bound,
 y is free

scope $\Rightarrow \exists x (P(x) \wedge Q(x)) \vee \forall x R(x)$

Here, scope of $\exists x$ is the expression $P(x) \wedge Q(x)$, not to the rest.

* Logically Equivalence involving quantifiers: $s \equiv t \rightarrow$ two statements with
| predicates and quantifiers
| and have same truth val

TABLE 2 De Morgan's Laws for Quantifiers.

Negation	Equivalent Statement	When Is Negation True?	When False?
$\neg \exists x P(x)$	$\forall x \neg P(x)$	For every x , $P(x)$ is false.	There is an x for which $P(x)$ is true.
$\neg \forall x P(x)$	$\exists x \neg P(x)$	There is an x for which $P(x)$ is false.	$P(x)$ is true for every x .

e.g. $\neg \forall x (P(x) \rightarrow Q(x)) \equiv \exists x (P(x) \wedge \neg Q(x))$

here, $\neg \forall x (P(x) \rightarrow Q(x))$
 $\equiv \exists x \neg (P(x) \rightarrow Q(x))$ [De Morgan's Law for quantifiers]
 $\equiv \exists x \neg (\neg P(x) \vee Q(x))$ [$P \rightarrow Q \equiv \neg P \vee Q$]
 $\equiv \exists x \neg (\neg P(x)) \wedge \neg Q(x)$ [De Morgan's Law for propositional logic]
 $\equiv \exists x (P(x) \wedge \neg Q(x))$ [double negation law]

Nested Quantifiers

$$\underbrace{\forall x \exists y}_{\text{Nested quantifiers}} (x+y=0) \Rightarrow \text{Here, } \forall x Q(x)$$

$Q(x)$ is $\exists y P(x,y)$
 $P(x,y)$ is $x+y=0$

negating nested quantifiers

Nested quantifiers

$$\text{Order} \Rightarrow \forall x \forall y P(x,y) \equiv \forall y \forall x P(x,y)$$

$\exists y \forall x Q(x,y) \neq \forall x \exists y Q(x,y)$

↓
matters

TABLE 1 Quantifications of Two Variables.

Statement	When True?	When False?
$\forall x \forall y P(x, y)$ $\forall y \forall x P(x, y)$	$P(x, y)$ is true for every pair x, y .	There is a pair x, y for which $P(x, y)$ is false.
$\forall x \exists y P(x, y)$	For every x there is a y for which $P(x, y)$ is true.	There is an x such that $P(x, y)$ is false for every y .
$\exists x \forall y P(x, y)$	There is an x for which $P(x, y)$ is true for every y .	For every x there is a y for which $P(x, y)$ is false.
$\exists x \exists y P(x, y)$ $\exists y \exists x P(x, y)$	There is a pair x, y for which $P(x, y)$ is true.	$P(x, y)$ is false for every pair x, y .

$$\neg \forall x \exists y (xy=1)$$

$$\equiv \exists x \neg \exists y (xy=1)$$

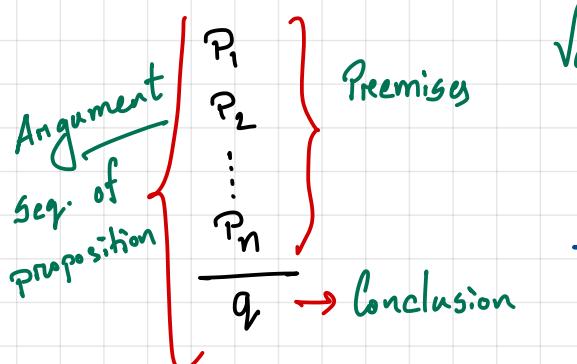
[De Morgan's Law]

$$\equiv \exists x \forall y \neg (xy=1)$$

[De Morgan's Law]

$$\equiv \exists x \forall y (xy \neq 1)$$

Rules of Inference



valid argument

$$\text{iff } (P_1 \wedge P_2 \wedge \dots \wedge P_n) \rightarrow q$$

is tautology

* Conclusion = true iff all premises = true

Example: premises $\neg p \wedge q$, $r \rightarrow p$, $\neg r \rightarrow s$, $s \rightarrow t$
 conclusion t

Step	Reason
1. $\neg p \wedge q$	premise
2. $\neg p$	simplification using (1)
3. $r \rightarrow p$	premise
4. $\neg r$	Modus tollens using (2), (3)
5. $\neg r \rightarrow s$	premise
6. s	Modus Ponens using (4), (5)
7. $s \rightarrow t$	premise
8. t	Modus ponens using (6), (7)

TABLE 1 Rules of Inference.

Rule of Inference	Tautology	Name
$\begin{array}{l} p \\ p \rightarrow q \\ \hline \therefore q \end{array}$	$(p \wedge (p \rightarrow q)) \rightarrow q$	Modus ponens
$\begin{array}{l} \neg q \\ p \rightarrow q \\ \hline \therefore \neg p \end{array}$	$(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$	Modus tollens
$\begin{array}{l} p \rightarrow q \\ q \rightarrow r \\ \hline \therefore p \rightarrow r \end{array}$	$((p \rightarrow q) \wedge (q \rightarrow r)) \rightarrow (p \rightarrow r)$	Hypothetical syllogism
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	$((p \vee q) \wedge \neg p) \rightarrow q$	Disjunctive syllogism
$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$	$p \rightarrow (p \vee q)$	Addition
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	$(p \wedge q) \rightarrow p$	Simplification
$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$	$((p) \wedge (q)) \rightarrow (p \wedge q)$	Conjunction
$\begin{array}{l} p \vee q \\ \neg p \vee r \\ \hline \therefore q \vee r \end{array}$	$((p \vee q) \wedge (\neg p \vee r)) \rightarrow (q \vee r)$	Resolution

* Resolution:

$$\frac{P \vee q}{\neg P \vee \text{H}} \rightarrow \text{Resolvent}$$

* let $q = \text{H}$ in Resolution tautology,

$$((P \vee q) \wedge (\neg P \vee q)) \rightarrow (q \vee q)$$

$$\equiv (P \vee q) \wedge (\neg P \vee q) \rightarrow q \vee q$$

* let $\text{H} = F$ in Resolution tautology,

$$((P \vee q) \wedge (\neg P \vee F)) \rightarrow (q \vee F)$$

$$\equiv (P \vee q) \wedge \neg P \rightarrow q \quad [\text{Disjunctive syllogism}]$$

*** To construct proofs in propositional logic using resolution as the only rule of inference, the hypotheses and conclusion must be expressed as clauses, where a clause is a disjunction of vars. or negations of these vars.

Replace propositional statement into clauses \Rightarrow e.g.

$$\begin{aligned} p \vee (q \wedge \text{H}) &\equiv (P \vee q) \wedge (P \vee \text{H}) \\ \neg(P \vee q) &\equiv \neg P \wedge \neg q \\ P \rightarrow q &\equiv \neg P \vee q \end{aligned}$$

2 clauses

* Fallacies:

Fallacy of affirming the conclusion $\Rightarrow ((P \rightarrow Q) \wedge Q) \rightarrow P$ is not tautology
 $[P = F, Q = T]$

Fallacy of denying the hypothesis $\Rightarrow ((P \rightarrow Q) \wedge \neg P) \rightarrow \neg Q$ is not tautology
 $[P = F, Q = T]$

* Rules of inference for quantified statements:

TABLE 2 Rules of Inference for Quantified Statements.

Rule of Inference	Name
$\frac{\forall x P(x)}{P(c)}$	Universal instantiation
$\frac{P(c) \text{ for an arbitrary } c}{\forall x P(x)}$	Universal generalization
$\frac{\exists x P(x)}{P(c) \text{ for some element } c}$	Existential instantiation
$\frac{P(c) \text{ for some element } c}{\exists x P(x)}$	Existential generalization

$P(c) \Rightarrow c$ is a particular member of the domain

element c that select must be arbitrary & not a specific



Show that the premises “A student in this class has not read the book,” and “Everyone in this class passed the first exam” imply the conclusion “Someone who passed the first exam has not read the book.”

Solution: Let $C(x)$ be “ x is in this class,” $B(x)$ be “ x has read the book,” and $P(x)$ be “ x passed the first exam.” The premises are $\exists x(C(x) \wedge \neg B(x))$ and $\forall x(C(x) \rightarrow P(x))$. The conclusion is $\exists x(P(x) \wedge \neg B(x))$. These steps can be used to establish the conclusion from the premises.

Step

1. $\exists x(C(x) \wedge \neg B(x))$
2. $C(a) \wedge \neg B(a)$
3. $C(a)$
4. $\forall x(C(x) \rightarrow P(x))$
5. $C(a) \rightarrow P(a)$
6. $P(a)$
7. $\neg B(a)$
8. $P(a) \wedge \neg B(a)$
9. $\exists x(P(x) \wedge \neg B(x))$

Reason

- Premise
- Existential instantiation from (1)
- Simplification from (2)
- Premise
- Universal instantiation from (4)
- Modus ponens from (3) and (5)
- Simplification from (2)
- Conjunction from (6) and (7)
- Existential generalization from (8)

Universal modus ponens

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

$P(a)$, where a is a particular element in the domain

$$\therefore Q(a)$$

Universal modus tollens

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

$\neg Q(a)$, where a is a particular element in the domain

$$\therefore \neg P(a)$$

Introduction to Proofs

a proof is a valid argument that establishes the truth of a mathematical statement.
formal proofs → where all steps were supplied & rules for each step in arg. given
informal proofs → where more than one rule of inference may be used in each step, where steps may be skipped, where axiom being assumed, rules of inference used are not explicitly stated.

Theorem \Rightarrow it is a statement that can be shown to be true.

↳ less important theorem \Rightarrow proposition

Proof \Rightarrow a proof is a valid argument that establishes truth of a theorem

Axiom/Postulates \Rightarrow which are statements we assume to be true.

it may be stated using primitive terms that do not req. defn.

Lemma \Rightarrow a less important theorem that is helpful in the proof of other results

Corollary \Rightarrow it is a theorem that can be established directly from a theorem that has been proved individually.

Conjecture \Rightarrow it is a statement that is being proposed to be true statement, usually on the basis of partial evidence, a heuristic arg., or the intuition of an expert.

* Methods of proving theorem:

Direct proofs \Rightarrow assume hypothesis of the conditional statement is true.

direct proofs lead from the premises of a theorem to conclusion

Proof by Contradiction

Indirect proofs

↳ Proof by Contraposition \Rightarrow use $p \rightarrow q \equiv \neg q \rightarrow \neg p$

Vacuous proof $\Rightarrow p \rightarrow q$ true, if p false \Rightarrow we have proof

Trivial proof $\Rightarrow p \rightarrow q$ true, if q true

Proof by Contradiction \Rightarrow assume $\neg p$ is true, and leads to a contradiction

↳ indirect proofs

Proofs of equivalence $\Rightarrow p \Leftrightarrow q$

Syntax: formulae A in logic, e.g.: w , $w \rightarrow s$

Semantics: which formulae A is true under which assignment φ , written $\varphi \models A$
 $\varphi := \{w \mapsto T, s \mapsto F\}$, then $\varphi \models w$ but $\varphi \not\models (w \rightarrow s)$

Entailment: $A \models B$, B are entailed by A, means for all φ with $\varphi \models A$, also $\varphi \models B$

Deduction: $A \vdash_c B$, statement B can be derived from A using a set of C inference rule.

* **Soundness:** when $A \vdash_c B$, we also have $A \models B$

* **Completeness:** when $A \models B$, we also have $A \vdash_c B$

Propositional Logic (PL⁰)

propositional vars, $\mathcal{V}_0 := \{P, Q, P^1, P^2, \dots\}$

connectives, $\Sigma_0 := \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$

$\neg A, A \wedge B, A \vee B, A \rightarrow B, A \leftrightarrow B$

atoms: propositional formulae without connectives

Complex! not atoms

* wff_0 - well-formed propositional formulae

Semantics of PL⁰ (Models)

model $M := \langle D_0, I \rangle$

Universe $D_0 = \{T, F\}$ Fixed

Interpretation I

$\vdash I(\neg) : D_0 \rightarrow D_0 ; T \rightarrow F, F \rightarrow T$

$I(\wedge) : D_0 \times D_0 \rightarrow D_0$

forl. assign. $\varphi : \mathcal{V}_0 \rightarrow D_0$

val. fun. $I\varphi : wff_0(\mathcal{V}_0) \rightarrow D_0$

$I\varphi(P) = \varphi(P)$

$I\varphi(\neg A) = I(\neg)(I\varphi(A))$

Truth table

Semantic Properties of Propositional Formulae

Definition 12.2.9. Let $M := \langle U, I \rangle$ be our model, then we call A

- ▷ true under φ (φ satisfies A) in M, iff $I\varphi(A) = T$
- ▷ false under φ (φ falsifies A) in M, iff $I\varphi(A) = F$
- ▷ satisfiable in M, iff $I\varphi(A) = T$ for some assignment φ
- ▷ valid in M, iff $M \models^\varphi A$ for all assignments φ
- ▷ falsifiable in M, iff $I\varphi(A) = F$ for some assignments φ
- ▷ unsatisfiable in M, iff $I\varphi(A) = F$ for all assignments φ

* valid formulae is called tautology

* Entailment: $A \models B$
A entails B

iff $I\varphi(B) = T$ for all φ with
 $I\varphi(A) = T$

* First order signature:

function constants: $\Sigma_K^f = \{f, g, h, \dots\}$ - denoting function on individuals

predicate constants: $\Sigma_K^p = \{P, Q, R, \dots\}$ - denoting relationship among individuals.

The formulae of PL^{Inq} are given by the following grammar

functions	f^k	\in	Σ_k^f
predicates	p^k	\in	Σ_k^p
terms	t	$::=$	X
			f^0
			$f^k(t_1, \dots, t_k)$
formulae	A	$::=$	$p^k(t_1, \dots, t_k)$
			$\neg A$
			$A_1 \wedge A_2$

The value function \mathcal{I} assigns values to formulae

- ▷ $\mathcal{I}(f(A^1, \dots, A^k)) := \mathcal{I}(f)(\mathcal{I}(A^1), \dots, \mathcal{I}(A^k))$
- ▷ $\mathcal{I}(p(A^1, \dots, A^k)) := T$, iff $(\mathcal{I}(A^1), \dots, \mathcal{I}(A^k)) \in \mathcal{I}(p)$
- ▷ $\mathcal{I}(\neg A) = \mathcal{I}(\neg)(\mathcal{I}(A))$ and $\mathcal{I}(A \wedge B) = \mathcal{I}(\wedge)(\mathcal{I}(A), \mathcal{I}(B))$

Natural Deduction Calculus: ND_o

Introduction

$$\frac{A \quad B}{A \wedge B} \wedge I$$

$$\frac{[A]'}{\frac{B}{A \rightarrow B}} \rightarrow I'$$

Assumption (A) , then derived B
 A can not be use again in proof.

Elimination

$$\frac{A \wedge B}{A} \wedge E_L \quad \frac{A \wedge B}{B} \wedge E_R$$

$$\frac{A \rightarrow B \quad A}{B} \rightarrow E$$

Axiom

$$\frac{}{A \vee \neg A} \text{TND}$$

$$\frac{A}{A \vee B} \vee I_L \quad \frac{B}{A \vee B} \vee I_R \quad A \vee B \quad \frac{[A]' \quad [B]'}{\frac{\vdots \quad \vdots}{C}} \vee E'$$

$$[A]^1$$

$$\frac{\vdots}{\neg A} \neg I'$$

$$\frac{\neg A \quad A}{F} FI$$

$$\frac{\neg \neg A}{A} \neg E$$

$$\frac{\bar{F}}{A} FE$$

ND_o (sequent style version) :

Deduction theorem for ND_o: $H, A \vdash_{\text{ND}_o} B$, iff $H \vdash_{\text{ND}_o} A \rightarrow B$

$$\begin{array}{c} \frac{}{\Gamma, A \vdash A} Ax \quad \frac{\Gamma \vdash B}{\Gamma, A \vdash B} \text{weaken} \quad \frac{}{\Gamma \vdash A \vee \neg A} \text{TND} \\ \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge I \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \wedge E_L \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B} \wedge E_R \\ \frac{\Gamma \vdash A \vee B}{\Gamma \vdash A \vee B} \vee I_l \quad \frac{\Gamma \vdash A \vee B}{\Gamma \vdash B} \vee I_r \quad \frac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C} \vee E \\ \frac{\Gamma, A \vdash B \Rightarrow I}{\Gamma \vdash A \Rightarrow B} \quad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B} \Rightarrow E \\ \frac{\Gamma, A \vdash F \neg I}{\Gamma \vdash \neg A} \quad \frac{\Gamma \vdash \neg \neg A}{\Gamma \vdash A} \neg E \\ \frac{\Gamma \vdash \neg A \quad \Gamma \vdash A}{\Gamma \vdash F} FI \quad \frac{\Gamma \vdash F}{\Gamma \vdash A} FE \end{array}$$

PL^{Inq} semantics:

$D_i \neq \emptyset$ of individuals,

$I: \Sigma_0^f \rightarrow D_i$ individual Const. \rightarrow individuals

$I: \Sigma_K^f \rightarrow D_i^K \rightarrow D_i$ funcⁿ or funcⁿ

$I: \Sigma_k^P \rightarrow P(D_i^K)$ predicate as arbitrary relation

PL^{Inq} & PL^o isomorphic ??

PL with Hilbert Calculus H^o ??

Linearized notation for sequent-style ND proofs

1. $\mathcal{H}_1 \vdash A_1 (\mathcal{J}_1)$
2. $\mathcal{H}_2 \vdash A_2 (\mathcal{J}_2)$
3. $\mathcal{H}_3 \vdash A_3 (\mathcal{J}_{31,2})$

correspond to,

$$\frac{\mathcal{H}_1 \vdash A_1 \quad \mathcal{H}_2 \vdash A_2}{\mathcal{H}_3 \vdash A_3} R$$

* **Unsatisfiability theorem**: $H \models A$ iff $H \cup \{\neg A\}$ is unsatisfiable

Normal Form:

▷ **Definition 13.1.6.**

A formula is in **conjunctive normal form (CNF)** if it is a conjunction of disjunctions of literals: i.e. if it is of the form $\bigwedge_{i=1}^n \bigvee_{j=1}^{m_i} l_{ij}$

▷ **Definition 13.1.7.**

A formula is in **disjunctive normal form (DNF)** if it is a disjunction of conjunctions of literals: i.e. if it is of the form $\bigvee_{i=1}^n \bigwedge_{j=1}^{m_i} l_{ij}$

Test Calculi: Tableaux

Formulae	$\xleftarrow{\text{atomic}}$ labeled formulae A^α , if $\alpha \in \{T, F\}$ $\xleftarrow{\text{complex}}$ labeled atom A^α
	$\xleftarrow{\text{pos if } \alpha = T}$ $\xleftarrow{\text{neg if } \alpha = F}$
opposite literal or partner	A^α, B^β if $\alpha \neq \beta$

inference rules of propositional tableau calculus T_0

$$\frac{(A \wedge B)^T}{\begin{array}{c} AT \\ BT \end{array}} T_0 \wedge \quad \frac{(A \wedge B)^F}{AF \mid BF} T_0 \vee \quad \frac{\neg A^T}{AF} T_0 \neg^T \quad \frac{\neg A^F}{A^T} T_0 \neg^F \quad \frac{\begin{array}{c} A^\alpha \\ A^\beta \end{array} \alpha \neq \beta}{\perp} T_0 \perp$$

tableau $\xleftarrow{\text{saturated}}$ iff no rule adds new materials & a branch closed, else open.
 $\xleftarrow{\text{closed}}$ iff all of its branches are closed

* T_0 -theorem/Derivability: A is a T_0 -theorem ($\vdash_{T_0} A$), iff there is a closed tableau with A^F at the root.

Derived rules of inference:

$$\frac{(A \Rightarrow B)^T}{\begin{array}{c} AF \\ BT \end{array}} \quad \frac{(A \Rightarrow B)^F}{\begin{array}{c} AT \\ BF \end{array}} \quad \frac{A^T}{\begin{array}{c} AT \\ BT \end{array}} \quad \frac{(A \vee B)^T}{\begin{array}{c} AT \\ AF \end{array}} \quad \frac{(A \vee B)^F}{\begin{array}{c} AF \\ BF \end{array}} \quad \frac{A \Leftrightarrow B^T}{\begin{array}{c} AT \\ BT \end{array}} \quad \frac{A \Leftrightarrow B^F}{\begin{array}{c} AT \\ BF \end{array}}$$

* **Soundness**: T_0 is sound, i.e. $\Phi \subseteq \text{wff}_0(\mathcal{I}_0)$ valid, if there is a closed tableau T for Φ^F

* **Completeness**: T_0 is complete, i.e. if $\Phi \subseteq \text{wff}_0(\mathcal{I}_0)$ is valid, then there is a closed tableau T for Φ^F

* **Termination**: T_0 terminates, i.e. every T_0 tableau becomes saturated after finitely many rule applications.

- tableau calculus basically computes disjunctive normal form (DNF), every branch is a disjunct that is conjunction of literals.

Resolution for PL:

inference rule for resolⁿ calculating R

$$\frac{P^T \vee A \quad P^F \vee B}{A \vee B} R$$

$P^T \quad P^F \quad \left. \begin{array}{l} P^T \\ P^F \end{array} \right\} \text{Cut Literals}$

* clause C

- ▷ **Definition 13.5.5 (Transformation into Clause Normal Form).** The **CNF transformation calculus CNF₀** consists of the following four **inference rules** on sets of labeled formulae.

$$\frac{C \vee (A \vee B)^T}{C \vee A^T \vee B^T} \quad \frac{C \vee (A \vee B)^F}{C \vee A^F; C \vee B^F} \quad \frac{C \vee \neg A^T}{C \vee A^F} \quad \frac{C \vee \neg A^F}{C \vee A^T}$$

derived rules of inference

$$\frac{C \vee (A \Rightarrow B)^T}{C \vee A^F \vee B^T} \quad \frac{C \vee (A \Rightarrow B)^F}{C \vee A^T; C \vee B^F} \quad \frac{C \vee (A \wedge B)^T}{C \vee A^T; C \vee B^T} \quad \frac{C \vee (A \wedge B)^F}{C \vee A^F \vee B^F}$$

Chap 14

Formal Systems: Syntax, Semantics, Entailment & Derivation in General

Let $\langle \mathcal{L}, \mathcal{K}, \models \rangle$ be a logical system, $M \in \mathcal{K}$ be a model and $A \in \mathcal{L}$ a formula, then we say that A is

- ▷ satisfied by M , iff $M \models A$.
- ▷ falsified by M , iff $M \not\models A$.
- ▷ satisfiable in \mathcal{K} , iff $M \models A$ for some $M \in \mathcal{K}$.
- ▷ valid in \mathcal{K} (write $\models M$), iff $M \models A$ for all $M \in \mathcal{K}$.
- ▷ falsifiable in \mathcal{K} , iff $M \not\models A$ for some $M \in \mathcal{K}$.
- ▷ unsatisfiable in \mathcal{K} , iff $M \not\models A$ for all $M \in \mathcal{K}$.

▷ Definition 14.0.12.

An inference rule \mathcal{I} is called **admissible** in a calculus \mathcal{C} , if the extension of \mathcal{C} by \mathcal{I} does not yield new theorems.

▷ **Definition 14.0.13.** An inference rule $\frac{A_1 \dots A_n}{C}$ is called **derivable** (or a **derived rule**) in a calculus \mathcal{C} , if there is a \mathcal{C} derivation $A_1, \dots, A_n \vdash_{\mathcal{C}} C$.

▷ **Observation 14.0.14.** Derivable inference rules are admissible, but not the other way around.

Chap 15

Propositional Reasoning: SAT Solvers

▷ **Definition 15.1.1.** The **SAT problem (SAT)**: Given a propositional formula A , decide whether or not A is **satisfiable**. We denote the class of all **SAT** problems with **SAT**

> The **SAT** problem was the first problem proved to be **NP-complete**!

> A is commonly assumed to be in **CNF**. This is **without loss of generality**, because any A can be transformed into a satisfiability-equivalent **CNF** formula (cf. chapter 12) in **polynomial time**.

SAT vs. CSP

▷ **Observation 15.1.3 (SAT: A kind of CSP).** **SAT** can be viewed as a **CSP** problem in which all **variable domains** are Boolean, and the **constraints** have unbounded arity.

▷ **Theorem 15.1.4 (Encoding CSP as SAT).** Given any **constraint network** \mathcal{C} , we can in low order **polynomial time** construct a **CNF** formula $A(\mathcal{C})$ that is **satisfiable** iff \mathcal{C} is **solvable**.

* Davis - Putnam (Logemann - Loveland) Procedure (DPLL):

- How to systematically test satisfiability
- a restricted form of resolution

DPLL (Vanilla Example) :

▷ Example 15.2.2 (UP and Splitting). Let $\Delta := (P^T \vee Q^T \vee R^F; P^F \vee Q^F; R^T; P^T \vee Q^F)$

1. UP Rule: $R \mapsto T$

$$P^T \vee Q^T; P^F \vee Q^F; P^T \vee Q^F$$

2. Splitting Rule:

$$\begin{array}{l} 2a. P \mapsto F \\ \quad Q^T; Q^F \end{array}$$

$$\begin{array}{l} 2b. P \mapsto T \\ \quad Q^F \end{array}$$

3a. UP Rule: $Q \mapsto T$

□

returning "unsatisfiable"

3b. UP Rule: $Q \mapsto F$

clause set empty

returning " $R \mapsto T, P \mapsto T, Q \mapsto F$ "

* DPLL $\hat{=}$ (restricted form of) resolution

* clause learning

??

▷ Definition 15.3.1 (Unit Resolution). Unit resolution (UR) is the test calculus consisting of the following inference rule:

$$\frac{C \vee P^\alpha \quad P^\beta \quad \alpha \neq \beta}{C} \text{ UR}$$

▷ Unit propagation $\hat{=}$ resolution restricted to cases where one parent is unit clause.

▷ Observation 15.3.2 (Soundness). UR is refutation sound. (since resolution is)

▷ Observation 15.3.3 (Completeness). UR is not refutation complete (alone).

Chap 16

First Order Predicate Logic

▷ Definition 16.2.2. A first-order signature consists of (all disjoint; $k \in \mathbb{N}$)

▷ connectives: $\Sigma^o = \{T, F, \neg, \vee, \wedge, \Rightarrow, \Leftrightarrow, \dots\}$ (functions on truth values)

▷ function constants: $\Sigma_k^f = \{f, g, h, \dots\}$ (functions on individuals)

▷ predicate constants: $\Sigma_k^p = \{p, q, r, \dots\}$ (relationships among individuals.)

▷ (Skolem constants: $\Sigma_k^{sk} = \{f_k^1, f_k^2, \dots\}$) (witness constructors; countably ∞)

| predicate Σ^p
| function Σ^f
| maps $x \rightarrow y$

PL' syntax

bound & free var.

▷ Definition 16.2.4. Terms: $A \in wff_i(\Sigma_i, V_i)$ (denote individuals)

▷ $V_i \subseteq wff_i(\Sigma_i, V_i)$,

▷ if $f \in \Sigma_k^f$ and $A^i \in wff_i(\Sigma_i, V_i)$ for $i \leq k$, then $f(A^1, \dots, A^k) \in wff_i(\Sigma_i, V_i)$.

▷ Definition 16.2.5. if Propositions: $A \in wff_o(\Sigma_i, V_i)$: (denote truth values)

▷ if $p \in \Sigma_k^p$ and $A^i \in wff_i(\Sigma_i, V_i)$ for $i \leq k$, then $p(A^1, \dots, A^k) \in wff_o(\Sigma_i, V_i)$,

▷ if $A, B \in wff_o(\Sigma_i, V_i)$ and $X \in V_i$, then $T, A \wedge B, \neg A, \forall X. A \in wff_o(\Sigma_i, V_i)$. \forall is a binding operator called the universal quantifier.

$\forall x P(x, y)$

| x bound
| y free

▷ Definition 16.2.6. We define the connectives $F, \vee, \Rightarrow, \Leftrightarrow$ via the abbreviations $A \vee B := \neg(\neg A \wedge \neg B)$, $A \Rightarrow B := \neg A \vee B$, $A \Leftrightarrow B := (A \Rightarrow B) \wedge (B \Rightarrow A)$, and $F := \neg T$. We will use them like the primary connectives \wedge and \neg

▷ Definition 16.2.7. We use $\exists X. A$ as an abbreviation for $\neg(\forall X. \neg A)$. \exists is a binding operator called the existential quantifier.

PL' Semantics:

- universe $D_i \neq \emptyset$ of individuals
- interpretation \mathcal{I} : $I: \Sigma_k^f \rightarrow D_i \rightarrow D_L$
- $I: \Sigma_k^P \rightarrow P(D_i^k)$

▷ Definition 16.2.17.

Given a model $\langle D, \mathcal{I} \rangle$, the value function \mathcal{I}_φ is recursively defined:
terms & propositions

* Assignment Extension!
 φ is var. assignment, then
 $\varphi, [a/x]$ is called extension
of φ with $[a/x]$

▷ $\mathcal{I}_\varphi: wff_\nu(\Sigma_\nu, V_\nu) \rightarrow D_\nu$ assigns values to terms.

▷ $\mathcal{I}_\varphi(X) := \varphi(X)$ and

▷ $\mathcal{I}_\varphi(f(A_1, \dots, A_k)) := \mathcal{I}(f)(\mathcal{I}_\varphi(A_1), \dots, \mathcal{I}_\varphi(A_k))$

▷ $\mathcal{I}_\varphi: wff_o(\Sigma_\nu, V_\nu) \rightarrow D_o$ assigns values to formulae:

▷ $\mathcal{I}_\varphi(T) = \mathcal{I}(T) = \top$,

▷ $\mathcal{I}_\varphi(\neg A) = \mathcal{I}(\neg)(\mathcal{I}_\varphi(A))$

▷ $\mathcal{I}_\varphi(A \wedge B) = \mathcal{I}(\wedge)(\mathcal{I}_\varphi(A), \mathcal{I}_\varphi(B))$

▷ $\mathcal{I}_\varphi(p(A_1, \dots, A_k)) := \top$, iff $\langle \mathcal{I}_\varphi(A_1), \dots, \mathcal{I}_\varphi(A_k) \rangle \in \mathcal{I}(p)$

▷ $\mathcal{I}_\varphi(\forall X.A) := \top$, iff $\mathcal{I}_{\varphi, [a/X]}(A) = \top$ for all $a \in D_\nu$.

* First Order Substitution:

Subs on term $[z/x]$

Subs on proposition: var capture $[f(x)/Y] (\forall x.P(x,y)) \equiv \forall x.P(x, f(x))$

\hookrightarrow free \rightarrow bound

First Order ND¹:

▷ **Definition 16.3.1 (New Quantifier Rules).** The first-order natural deduction calculus ND^1 extends ND_0 by the following four rules:

$$\frac{\mathbf{A}}{\forall X.A} \forall I^* \quad \frac{\forall X.A}{[B/X](A)} \forall E$$

$$[[c/X](A)]^1$$

$$\frac{[B/X](A) \exists I}{\exists X.A} \quad \frac{\exists X.A \quad \vdots \quad c \in \Sigma_0^{sk} \text{ new}}{C \quad C} \exists E^1$$

First-order ND¹ in sequent form:

▷ **Definition 16.3.3 (New Quantifier Rules).** The inference rules of the first-order sequent calculus ND_F^1 consist of those from ND_F^0 plus the following quantifier rules:

$$\frac{\Gamma \vdash A \quad X \notin \text{free}(\Gamma)}{\Gamma \vdash \forall X.A} \forall I \quad \frac{\Gamma \vdash \forall X.A}{\Gamma \vdash [B/X](A)} \forall E$$

$$\frac{\Gamma \vdash [B/X](A) \exists I}{\Gamma \vdash \exists X.A} \quad \frac{\Gamma \vdash \exists X.A \quad \Gamma, [c/X](A) \vdash C \quad c \in \Sigma_0^{sk} \text{ new}}{\Gamma \vdash C} \exists E$$

▷ **Definition 16.3.5.** For the calculus of natural deduction with equality (ND_E^1) we add the following two rules to ND^1 to deal with equality:

$$\frac{}{A = A} = I \quad \frac{A = B \quad C[A]_p = E}{[B/p]C} = E$$

where $C[A]_p$ if the formula C has a subterm A at position p and $[B/p]C$ is the result of replacing that subterm with B .

Automated Theorem Proving in F₀ Logic

F₀ Standard Tableaux (\mathcal{T}_1)

▷ **Definition 17.1.8.** The standard tableau calculus (\mathcal{T}_1) extends \mathcal{T}_0 (propositional tableau calculus) with the following quantifier rules:

$$\frac{(\forall X.A)^T \quad C \in \text{cuff}_{\nu}(\Sigma)}{([C/X](A))^T} \mathcal{T}_1 \forall \quad \frac{(\forall X.A)^F \quad c \in \Sigma_0^{sk} \text{ new}}{([c/X](A))^F} \mathcal{T}_1 \exists$$

Free var. Tableaux (\mathcal{T}_1^f)

▷ **Definition 17.1.9.** The free variable tableau calculus (\mathcal{T}_1^f) extends \mathcal{T}_0 (propositional tableau calculus) with the quantifier rules:

$$\frac{(\forall X.A)^T \quad Y \text{ new}}{([Y/X](A))^T} \mathcal{T}_1^f \forall \quad \frac{(\forall X.A)^F \quad \text{free}(\forall X.A) = \{X^1, \dots, X^k\} \quad f \in \Sigma_k^{sk} \text{ new}}{([f(X^1, \dots, X^k)/X](A))^F} \mathcal{T}_1^f \exists$$

and generalizes its cut rule $\mathcal{T}_0 \perp$ to:

$$\frac{\begin{array}{c} A^\alpha \\ B^\beta \end{array} \quad \alpha \neq \beta \quad \sigma(A) = \sigma(B)}{\perp : \sigma} \mathcal{T}_1^f \perp$$

$\mathcal{T}_1^f \perp$ instantiates the whole tableau by σ .

Derivable quantifier rules in \mathcal{T}_1^f

$$\frac{(\exists X.A)^T \quad \text{free}(\forall X.A) = \{X^1, \dots, X^k\} \quad f \in \Sigma_k^{sk} \text{ new}}{([f(X^1, \dots, X^k)/X](A))^T}$$

$$\frac{(\exists X.A)^F \quad Y \text{ new}}{([Y/X](A))^F}$$

First Order Unification :

Start, close left branch	use $\mathcal{T}_1^f \forall$ again (right branch)
$((p(a) \vee p(b)) \Rightarrow (\exists.p(x)))^F$ $(p(a) \vee p(b))^T$ $(\exists.x.p(x))^F$ $(\forall.x.\neg p(x))^T$ $\neg p(y)^T$ $p(y)^F$ $p(a)^T \quad \quad p(b)^T$ $\perp : [a/y] \quad \quad \perp : [b/z]$	$((p(a) \vee p(b)) \Rightarrow (\exists.p(x)))^F$ $(p(a) \vee p(b))^T$ $(\exists.x.p(x))^F$ $(\forall.x.\neg p(x))^T$ $\neg p(a)^T$ $p(a)^F$ $\perp : [a/y]$ $p(b)^T$ $\neg p(z)^T$ $p(z)^F$ $\perp : [b/z]$

Example of \mathcal{T}_1^f

use again

use again

▷ **Definition 17.1.12.** For given terms A and B , unification is the problem of finding a substitution σ , such that $\sigma(A) = \sigma(B)$.

unification algo :

$$\frac{\mathcal{E} \wedge f(A^1, \dots, A^n) = ? f(B^1, \dots, B^n)}{\mathcal{E} \wedge A^1 = ? B^1 \wedge \dots \wedge A^n = ? B^n} \mathcal{U}_{\text{dec}}$$

$$\frac{\mathcal{E} \wedge A = ? A}{\mathcal{E}} \mathcal{U}_{\text{triv}}$$

$$\frac{\mathcal{E} \wedge X = ? A \quad X \notin \text{free}(\mathcal{E}) \quad X \in \text{free}(\mathcal{E})}{[A/X](\mathcal{E}) \wedge X = ? A} \mathcal{U}_{\text{elim}}$$

unification e.g.

$f(g(X, X), h(a)) = ? f(g(a, Z), h(Z))$ $\frac{g(X, X) = ? g(a, Z) \wedge h(a) = ? h(Z)}{X = ? a \wedge X = ? Z \wedge h(a) = ? h(Z)}$ $\frac{X = ? a \wedge X = ? Z \wedge h(a) = ? h(Z)}{X = ? a \wedge X = ? Z \wedge a = ? Z}$ $\frac{X = ? a \wedge X = ? Z \wedge a = ? Z}{X = ? a \wedge Z = ? a \wedge a = ? a}$ $\frac{X = ? a \wedge Z = ? a \wedge a = ? a}{X = ? a \wedge Z = ? a}$	$f(g(X, X), h(a)) = ? f(g(b, Z), h(Z))$ $\frac{g(X, X) = ? g(b, Z) \wedge h(a) = ? h(Z)}{X = ? b \wedge X = ? Z \wedge h(a) = ? h(Z)}$ $\frac{X = ? b \wedge X = ? Z \wedge h(a) = ? h(Z)}{X = ? b \wedge X = ? Z \wedge a = ? Z}$ $\frac{X = ? b \wedge X = ? Z \wedge a = ? Z}{X = ? b \wedge b = ? Z \wedge a = ? Z}$ $\frac{X = ? b \wedge b = ? Z \wedge a = ? Z}{X = ? b \wedge Z = ? b \wedge a = ? b}$
MGU: $[a/X], [a/Z]$	$a = ? b$ not unifiable

first order resolution (à CNF)

▷ **Definition 17.2.1.** The first-order CNF calculus CNF_1 is given by the inference rules of CNF_0 extended by the following quantifier rules:

$$\frac{(\forall X.A)^T \vee C \ Z \notin (\text{free}(A) \cup \text{free}(C))}{([Z/X](A))^T \vee C}$$

$$\frac{(\forall X.A)^F \vee C \ \{X_1, \dots, X_k\} = \text{free}(\forall X.A) \ f \in \Sigma_k^{sk} \text{ new}}{([f(X_1, \dots, X_k)/X](A))^F \vee C}$$

* Horn clause is a clause with at most one positive literal

$\text{CNF}_1(\Phi)$ is the set of all clauses that can be derived from Φ .

▷ **Definition 17.2.2 (First-Order Resolution Calculus).** First-order resolution (\mathcal{R}_1) is a test calculus that manipulates formulae in conjunctive normal form. \mathcal{R}_1 has two inference rules:

$$\frac{A^T \vee C \ B^F \vee D \ \sigma = \text{mgu}(A, B)}{(\sigma(C)) \vee (\sigma(D))} \quad \frac{A^\alpha \vee B^\alpha \vee C \ \sigma = \text{mgu}(A, B)}{(\sigma(A)) \vee (\sigma(C))}$$

Derived rules for FD CNF

$$\frac{(\exists X.A)^T \vee C \ \{X_1, \dots, X_k\} = \text{free}(\exists X.A) \ f \in \Sigma_k^{sk} \text{ new}}{([f(X_1, \dots, X_k)/X](A))^T \vee C}$$

$$\frac{(\exists X.A)^F \vee C \ Z \notin (\text{free}(A) \cup \text{free}(C))}{([Z/X](A))^F \vee C}$$

three principle modes of inference

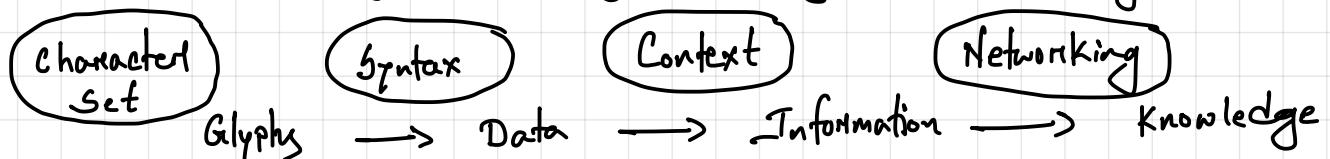
- ▶ **Definition 3.22.** Deduction $\hat{=}$ knowledge extension
- ▶ **Example 3.23.** $\frac{\text{rains} \Rightarrow \text{wet_street} \ \text{rains}}{\text{wet_street}} D$
- ▶ **Definition 3.24.** Abduction $\hat{=}$ explanation
- ▶ **Example 3.25.** $\frac{\text{rains} \Rightarrow \text{wet_street} \ \text{wet_street}}{\text{rains}} A$
- ▶ **Definition 3.26.** Induction $\hat{=}$ learning general rules from examples
- ▶ **Example 3.27.** $\frac{\text{wet_street} \ \text{rains}}{\text{rains} \Rightarrow \text{wet_street}} I$

chap 18

Knowledge Representation

& the Semantic Web

Intro of KR: knowledge is the information necessary to support intelligent reasoning! knowledge appears in layers,



* evaluation criteria for KR approaches:

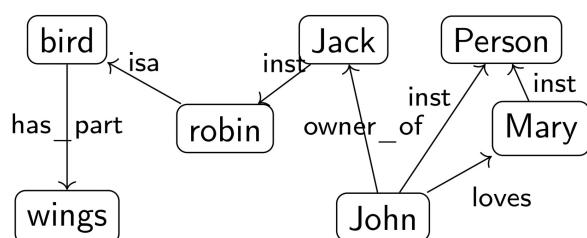
- Expressive adequacy
- Reasoning efficiency
- Primitives
- Meta representation
- Completeness

Semantic Networks: [example of KR]

semantic network is a directed graph for representing knowledge

- nodes : objects & concepts (classes of obj.)
- edges : relation between nodes

A semantic network for birds and persons:



links

'isa' an inclusion or isa link
 'inst' instance or inst link

▷ Definition 18.1.7 (Inference in Semantic Networks). We call all link labels except "inst" and "isa" in a semantic network relations.

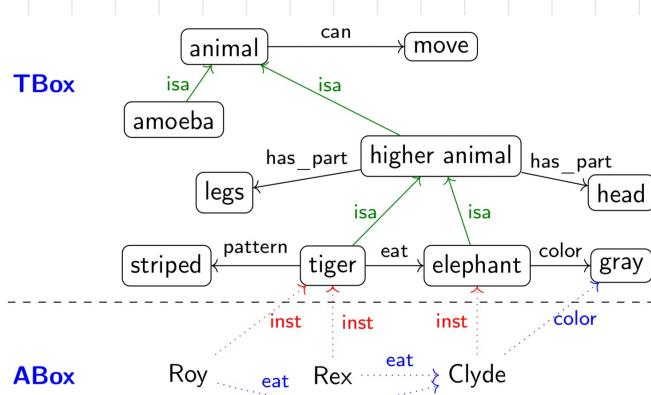
Let N be a semantic network and R a relation in N such that $A \xrightarrow{\text{isa}} B \xrightarrow{R} C$ or $A \xrightarrow{\text{inst}} B \xrightarrow{R} C$, then we can derive a relation $A \xrightarrow{R} C$ in N .

The process of deriving new concepts and relations from existing ones is called inference and concepts/relations that are only available via inference implicit (in a semantic network).

subgraph of semantic network N
 spanned by isa links & relations
 between concepts
 → terminology / TBox

subgraph spanned by inst links
 and relations between objects
 — assertions / ABox

* Semantic Web



Propositional Logic as Set Description Language:

► **Definition 18.2.1.** Let PL_{DL}^0 be given by the following grammar for the PL_{DL}^0 concepts.

$$\mathcal{L} ::= C \mid \top \mid \perp \mid \overline{\mathcal{L}} \mid \mathcal{L} \sqcap \mathcal{L} \mid \mathcal{L} \sqcup \mathcal{L} \mid \mathcal{L} \sqsubseteq \mathcal{L} \mid \mathcal{L} \equiv \mathcal{L}$$

i.e. PL_{DL}^0 formed from

- ▷ atomic (≡ propositional variables)
- ▷ concept intersection (\sqcap) (≡ conjunction \wedge)
- ▷ concept complement (\cdot) (≡ negation \neg)
- ▷ concept union (\sqcup), subsumption (\sqsubseteq), and equality (\equiv) defined from these. (≡ \vee , \Rightarrow , and \Leftrightarrow)

Definition 2.2 (Formal Semantics).

Let \mathcal{D} be a given set (called the domain) and $\varphi: \mathcal{V}_0 \rightarrow \mathcal{P}(\mathcal{D})$, then we define

- $[\![P]\!] := \varphi(P)$, (remember $\varphi(P) \subseteq \mathcal{D}$).
- $[\![A \sqcap B]\!] := [\![A]\!] \cap [\![B]\!]$ and $[\![A]\!] := \mathcal{D} \setminus [\![A]\!]$...

$$[\![\top]\!] = [p] \cup [\!\bar{p}\!] = [p] \cup (\mathcal{D} \setminus [p]) = \mathcal{D}$$

Analogously: $[\![\perp]\!] = \emptyset$

Translation into PL: $- \text{fo}(x) \rightarrow$ the translation into F0 logic w.r.t x

Definition	Comment
$\overline{p}^{\text{fo}(x)} := p(x)$	
$\overline{A}^{\text{fo}(x)} := \neg A$	
$\overline{A \sqcap B}^{\text{fo}(x)} := \overline{A}^{\text{fo}(x)} \wedge \overline{B}^{\text{fo}(x)}$	\wedge vs. \sqcap
$\overline{A \sqcup B}^{\text{fo}(x)} := \overline{A}^{\text{fo}(x)} \vee \overline{B}^{\text{fo}(x)}$	\vee vs. \sqcup
$\overline{A \sqsubseteq B}^{\text{fo}(x)} := \overline{A}^{\text{fo}(x)} \Rightarrow \overline{B}^{\text{fo}(x)}$	\Rightarrow vs. \sqsubseteq
$\overline{A = B}^{\text{fo}(x)} := \overline{A}^{\text{fo}(x)} \Leftrightarrow \overline{B}^{\text{fo}(x)}$	\Leftrightarrow vs. $=$
$\overline{A}^{\text{fo}} := (\forall x. \overline{A}^{\text{fo}(x)})$	for formulae

Ontologies & Description Logics:

↳ world description

semantic networks

$\text{PL}_{\text{DL}}^0, \text{PL}^1$

⇒ ontology format example

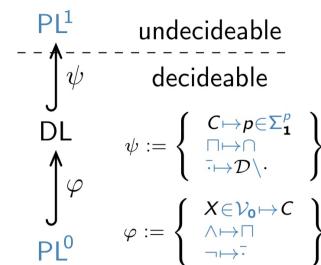
► **Definition 2.10.** An ontology consists of a logical system $\langle \mathcal{L}, \mathcal{K}, \models \rangle$ and concept axioms (expressed in \mathcal{L}) about

- individuals: concrete instances of object in the domain,
- concepts: classes of individuals that share properties and aspects, and
- relations: ways in which concept and individuals can be related to one another.

► **Definition 2.14.** A description logic is a formal system for talking about collections of objects and their relations that is at least as expressive as PL^0 with set-theoretic semantics and offers individuals and relations.

A description logic has the following four components:

- a formal language \mathcal{L} with logical constants $\sqcap, \neg, \sqcup, \sqsubseteq$, and \equiv ,
- a set-theoretic semantics $\langle \mathcal{D}, [\![\cdot]\!] \rangle$,
- a translation into first-order logic that is compatible with $\langle \mathcal{D}, [\![\cdot]\!] \rangle$, and
- a calculus for \mathcal{L} that induces a decision procedure for \mathcal{L} -satisfiability.



TBox & ABox

► **Definition 2.15.** Given a description logic \mathcal{D} , a \mathcal{D} ontology consists of

- a terminology (or TBox): concepts and roles and a set of concept axioms that describe them, and
- sassertion (or ABox): a set of individuals and statements about concept membership and role relationships for them.

* TBox is a finite set of concept defⁿ and concept axioms. It is called acyclic, iff it does not contain recursive defⁿ.

▷ Definition 18.2.20. A formula A is called **normalized** wrt. an TBox \mathcal{T} , iff it does not contain **concepts** defined in \mathcal{T} . (convenient)

▷ Definition 18.2.21 (**Algorithm**). (for arbitrary DLs)
Input: A formula A and a TBox \mathcal{T} .

- ▷ While [A contains concept c and \mathcal{T} a concept definition $c = C$]
 - ▷ substitute c by C in A.

Description logic & Inference!

▷ Definition 18.2.23. Ontology systems employ three main reasoning services:

- ▷ **Consistency test**: is a concept definition satisfiable?
- ▷ **Subsumption test**: does a concept subsume another?
- ▷ **Instance test**: is an individual an example of a concept?

Simple description logic : ALC

Syntax of ALC :

- Concepts : person, woman, man — etc
- Special Concepts ↗ top concept \top (for "true" or "all")
↘ bottom concept \perp (for "false" or "none")
- Roles: binary relations | has-child, loves, ... etc.

▷ Definition 18.3.7 (Grammar). $F_{ALC} ::= C \mid \top \mid \perp \mid \overline{F_{ALC}} \mid F_{ALC} \sqcap F_{ALC} \mid F_{ALC} \sqcup F_{ALC} \mid \exists R.F_{ALC} \mid \forall R.F_{ALC}$

* all/only \Rightarrow use \forall

* at least \Rightarrow use \exists

class inclusion! Prof. \sqsubseteq FacultyMember

class equivalence! Prof. \equiv FacultyMember

e.g. man = person $\sqcap \exists \text{hab-chrom. Y-chrom}$

definiendum $\xrightarrow{\text{primitive}}$ primitive concept

TBox Normalization in ALC! \rightarrow for only non-acyclic

▷ Definition 3.15. We call an ALC formula φ **normalized** wrt. a set of **concept definitions**, iff all concept names occurring in φ are **primitive**.

▷ Definition 3.16. Given a set \mathcal{D} of **concept definitions**, **normalization** is the process of replacing in an ALC formula φ all occurrences of **definienda** in \mathcal{D} with their **definientia**.

Semantics of ALC :

▷ Definition 18.3.21. A **model** for ALC is a pair $\langle \mathcal{D}, [\cdot] \rangle$, where \mathcal{D} is a non-empty set called the **domain** and $[\cdot]$ a mapping called the **interpretation**, such that

Op.	formula semantics
	$[c] \subseteq \mathcal{D} = [\top] \quad [\perp] = \emptyset \quad [r] \subseteq \mathcal{D} \times \mathcal{D}$
\neg	$[\neg \varphi] = [\varphi] = \mathcal{D} \setminus [\varphi]$
\sqcap	$[\varphi \sqcap \psi] = [\varphi] \sqcap [\psi]$
\sqcup	$[\varphi \sqcup \psi] = [\varphi] \sqcup [\psi]$
$\exists R.$	$[\exists R.\varphi] = \{x \in \mathcal{D} \mid \exists y. (x, y) \in [R] \text{ and } y \in [\varphi]\}$
$\forall R.$	$[\forall R.\varphi] = \{x \in \mathcal{D} \mid \forall y. \text{if } (x, y) \in [R] \text{ then } y \in [\varphi]\}$

A_{LC} into PL¹:

▷ **Definition 18.3.22.** The translation of A_{LC} into PL¹ extends the one from ?? by the following quantifier rules:

$$\overline{\forall R.\varphi}^{fo(x)} := (\forall y.R(x, y) \Rightarrow \overline{\varphi}^{fo(y)}) \quad \overline{\exists R.\varphi}^{fo(x)} := (\exists y.R(x, y) \wedge \overline{\varphi}^{fo(y)})$$

A_{LC} Identities:

1	$\overline{\exists R.\varphi} = \forall R.\overline{\varphi}$	3	$\overline{\forall R.\varphi} = \exists R.\overline{\varphi}$
2	$\overline{\forall R.(\varphi \sqcap \psi)} = \forall R.\varphi \sqcap \forall R.\psi$	4	$\overline{\exists R.(\varphi \sqcup \psi)} = \exists R.\varphi \sqcup \exists R.\psi$

▷ Proof of 1

$$\begin{aligned} [[\exists R.\varphi]] &= \mathcal{D} \setminus [\exists R.\varphi] = \mathcal{D} \setminus \{x \in \mathcal{D} \mid \exists y.((x, y) \in [R]) \text{ and } (y \in [\varphi])\} \\ &= \{x \in \mathcal{D} \mid \text{not } \exists y.((x, y) \in [R]) \text{ and } (y \in [\varphi])\} \\ &= \{x \in \mathcal{D} \mid \forall y. \text{if } ((x, y) \in [R]) \text{ then } (y \notin [\varphi])\} \\ &= \{x \in \mathcal{D} \mid \forall y. \text{if } ((x, y) \in [R]) \text{ then } (y \in (\mathcal{D} \setminus [\varphi]))\} \\ &= \{x \in \mathcal{D} \mid \forall y. \text{if } ((x, y) \in [R]) \text{ then } (y \in [\overline{\varphi}])\} \\ &= [[\forall R.\overline{\varphi}]] \end{aligned}$$

* A_{LC} with Assertions about individuals (ABox)

Definition 3.25. We define the assertions for A_{LC}

- ▶ $a:\varphi$
- ▶ $a R b$

assertions make up the ABox in A_{LC}.

(a is a φ)
(a stands in relation R to b)

individuals
↑
 $a:\varphi$ → concept

Definition 3.26. Let $\langle \mathcal{D}, [\cdot] \rangle$ be a model for A_{LC}, then we define

- ▶ $[a:\varphi] = T$, iff $[a] \in [\varphi]$, and
- ▶ $[a R b] = T$, iff $([a], [b]) \in [R]$.

Definition 3.27. We extend the PL¹ translation of A_{LC} to A_{LC} assertions:

- ▶ $\overline{a:\varphi}^{fo(a)} := \overline{\varphi}^{fo(a)}$, and
- ▶ $\overline{a R b}^{fo} := R(a, b)$.

Inference for A_{LC}:

Definition 3.28. The A_{LC} tableau calculus \mathcal{T}_{LC} acts on assertions

- ▶ $x:\varphi$ (x inhabits concept φ)
- ▶ $x R y$ (x and y are in relation R)

where φ is a normalized A_{LC} concept in negation normal form with the following rules:

$$\frac{x:c}{\perp} \mathcal{T}_\perp \quad \frac{x:\varphi \sqcap \psi}{x:\varphi \quad x:\psi} \mathcal{T}_\sqcap \quad \frac{x:\varphi \sqcup \psi}{x:\varphi \quad | \quad x:\psi} \mathcal{T}_\sqcup \quad \frac{x:\forall R.\varphi}{x R y \quad y:\varphi} \mathcal{T}_\forall \quad \frac{x:\exists R.\varphi}{x R y \quad y:\varphi} \mathcal{T}_\exists$$

We study the following properties of a tableau calculus \mathcal{C} :

- ▶ **Termination:** there are no infinite sequences of rule applications.
- ▶ **Correctness:** If φ is satisfiable, then \mathcal{C} terminates with an open branch.
- ▶ **Completeness:** If φ is in unsatisfiable, then \mathcal{C} terminates and all branches are closed.
- ▶ **Complexity** of the algorithm (time and space complexity).

Negation Normal Form (NNF)

A_{LC} formula is in NNF
iff \neg is only applied to
concept names.

example	by rule
$\exists R.(\forall S.e \sqcap \forall S.d)$	$\exists R.\varphi \mapsto \forall R.\overline{\varphi}$
$\mapsto \forall R.\forall S.e \sqcap \forall S.d$	$\varphi \sqcap \psi \mapsto \overline{\varphi} \sqcup \overline{\psi}$
$\mapsto \forall R.(\forall S.e \sqcup \forall S.d)$	$\forall R.\varphi \mapsto \forall R.\overline{\varphi}$
$\mapsto \forall R.(\exists S.e \sqcup \forall S.d)$	$\overline{\varphi} \mapsto \varphi$
$\mapsto \forall R.(\exists S.e \sqcap \forall S.d)$	

Correctness

Lemma 3.35. If φ satisfiable, then \mathcal{T}_{AC} terminates on $x:\varphi$ with open branch.

Proof: Let $\mathcal{M} = \langle \mathcal{D}, \llbracket \cdot \rrbracket \rangle$ be a model for φ and $w \in \llbracket \varphi \rrbracket$.

1. We define $\llbracket x \rrbracket := w$ and $\mathcal{I} \models x R y \iff \langle x, y \rangle \in \llbracket R \rrbracket$
2. This gives us $\mathcal{M} \models (x:\varphi)$ (base case)
3. If the branch is satisfiable, then either
 - no rule applicable to leaf, (open branch)
 - or rule applicable and one new branch satisfiable. (inductive case)
4. There must be an open branch. (by termination)

Completeness

Lemma 3.36. Open saturated tableau branches for φ induce models for φ .

Proof: construct a model for the branch and verify for φ

1. Let \mathcal{B} be an open saturated branch

- we define

$$\begin{aligned}\mathcal{D} &:= \{x \mid x:\psi \in \mathcal{B} \text{ or } z R x \in \mathcal{B}\} \\ \llbracket c \rrbracket &:= \{x \mid x:c \in \mathcal{B}\} \\ \llbracket R \rrbracket &:= \{\langle x, y \rangle \mid x R y \in \mathcal{B}\}\end{aligned}$$

- well-defined since never $x:c, x:\bar{c} \in \mathcal{B}$ (otherwise \mathcal{T}_{\perp} applies)
 - \mathcal{M} satisfies all constraints $x:c, x:\bar{c}$ and $x R y$, (by construction)
2. $\mathcal{M} \models (y:\psi)$, for all $y:\psi \in \mathcal{B}$ (on $k = \text{size}(\psi)$ next slide)
 3. $\mathcal{M} \models (x:\varphi)$.

ABox, Instance testing :

Definition 3.44. An **instance test** computes whether given an \mathcal{ALC} ontology an individual is a member of a given class.

Example 3.45 (An Ontology).

TBox (terminological Box)	ABox (assertional Box, data base)
woman = person \sqcap has_Y	tony:person
man = person \sqcap has_Y	tony:has_Y

This entails: tony:man (Tony is a man).

- **Definition 3.46.** **Realization** is the computation of all instance relations between ABox objects and TBox concepts.