Propositional Resolution
Introduction

(Nilsson Book Handout)

Professor Anita Wasilewska
CSE 352  Artificial Intelligence
Propositional Resolution
Part 1
SYNTAX “dictionary”

**Literal** – any *propositional* VARIABLE $a$ or negation of a variable $\neg a$, for $a \in \text{VAR}$

**Example:** variables: $a, b, c$ .... negation of variables: $\neg a, \neg b, -d$ ...

**Positive Literal:** any variable $a \in \text{VAR}$

**Clause** – any *finite set* of *literals*

**Example:** $C_1, C_2, C_3$ are clauses where

$C_1 = \{ a, b \} , \ C_2 = \{ a, \neg c \} , \ C_3 = \{ a, \neg a, \ldots, a_k \}$
Syntax “Dictionary”

Empty Clause:  \{\}  is an empty set i.e. a clause without elements

Finite set of clauses

\textbf{CL} = \{ C1, ...., Cn \}

Example

\textbf{CL} = \{\{a\}, \{ \}, \{ b, \neg a \}, \{c, \neg d\}\}
Semantics – Interpretation of Clauses

• Think **semantically** of a clause
  
  **C = \{ a_1, \ldots, a_n \}** as **disjunction**, i.e.
  
  C is logically equivalent to
  
  \[ a_1 \lor a_2 \lor \ldots \lor a_n \quad a_i \in \text{Literal} \]

• **Formally** – given a truth assignment \( v : \text{VAR} \rightarrow \{0, 1\} \)
  
  we extended it to set of all **CLAUSES** \( \text{CL} \) as follows:

  \[
  v^* : \text{CL} \rightarrow \{0, 1\} \\
  v^*(C) = v^*(a_1) \lor \ldots \lor v^*(a_n)
  \]

  for any clause \( C \) in \( \text{CL} \), where

  \[
  0 – \text{False, \quad 1 – True}
  \]

  **Shorthand** : \( v^* = v \)
Satisfiability, Model, Tautology

Example: let $v : \text{VAR} \rightarrow \{0, 1\}$ be such that

- $v(a) = 1$, $v(b) = 1$, $v(c) = 0$

and let $C = \{a, \neg b, c, \neg a\}$

We evaluate:

$v(C) = v(a) \lor \neg v(b) \lor v(c) \lor \neg v(a) = 1 \lor 0 \lor 0 \lor 1 = 1$

**OBSERVE** that $v(C) = 1$ for all $v$, i.e. the clause $C = \{a, \neg b, c, \neg a\}$ is a **Tautology**
Satisfiability, Model, Tautology

Definitions

1. For any clause $C$, and any truth assignment $v$ we write $v \models C$ and say that $v$ satisfies $C$ iff $v(C) = 1$

2. Any $v$ such that $v \models C$ is called a MODEL for $C$

3. A clause $C$ is satisfiable iff it has a MODEL, i.e.

   $C$ is satisfiable iff there is a $v$ such that $v \models C$

4. A clause $C$ is a tautology iff $v \models C$ for all $v$, i.e. all truth assignments $v$ are models for $C$
Notations

• $a, a, a$ is a finite sequence of 3 elements
• $\{a, a, a\} = \{a\}$ is a finite set
• $a, b, c \neq b, a, c$ are different sequences
• $\{a, b, c\} = \{b, a, c\}$ are the same sets
• $\{a, a, b, c\}$ is a multi-set (if needed)
Sets of Clauses CL

DEFINITIONS

1. A clause $C$ is **unsatisfiable** iff it has no **MODEL**
i.e. $v(C) = 0$ for all truth assignments $v$

   **Remark:** the empty clause $\emptyset$ is the only unsatisfiable clause

Let $CL = \{ C_1, \ldots, C_n \}$ be a **finite set of clauses**.

2. We extended $v : \text{VAR} \rightarrow \{0, 1\}$ to any set of clauses $CL$

   \[ v(\{ C_1, \ldots, C_n \}) = v(C_1) \land \ldots \land v(C_n) \]

A finite set of clauses $CL$ is semantically equivalent to a conjunction of all clauses in the set $CL$
Unsatisfiability

Definitions

1. A set of clauses $\text{CL}$ is satisfiable iff it has a model, i.e. iff $\exists v \ v(\text{CL}) = 1$

2. A set of clauses $\text{CL}$ is unsatisfiable iff it does not have a model, i.e. iff $\forall v \ v(\text{CL}) = 0$.

Remark:
If $\{\} \in \text{CL}$ then $\text{CL}$ is unsatisfiable
Unsatisfiability

Consider a set of clauses

\[ \text{CL} = \{\{a\}, \{a,b\}, \{\neg b\}\} \]

\text{CL} is \textbf{satisfiable} because any \( v \), such that \( v(a) = 1, v(b) = 0 \) is a \textbf{model} for \text{CL}

Check: \( v(\text{CL}) = 1 \land (1 \lor 0) \land 1 = 1 \)

\textbf{FACT:} When \( \{a\} \) and \( \{\neg a\} \) are in \text{CL},
then the set \text{CL} is \textbf{unisfiable}

Remember: \( (a \land \neg a) \) is a contradiction
Syntax and Semantics

- Example:
  - \(C_1 = \{a, b, \neg c\}, \ C_2 = \{c, a\}\) - syntax
  - \(C_1 = a \cup b \cup \neg c\) - semantics
  - \(C_2 = c \cup a\) - semantics

- \(CL = \{C_1, C_2\} = \{\{a, b, \neg c\}, \{c, a\}\}\) – syntax

\(CL = (a \cup b \cup \neg c) \land (c \cup a)\) - semantics
Syntax and Semantics

Definitions:

CL is **satisfiable** iff there is \( v \), such that \( v(\ CL\ ) = 1 \)

CL is **unsatisfiable** iff for all \( v \), \( v(CL) = 0 \)

- \( CL = \{ C1,C2, ...... ,Cn \} \) - syntax
- \( CL = C1 \land ...... \land Cn \) - semantics
Semantical Decidability

• A statement:
• “A finite set $\text{CL}$ of clauses is/ is not satisfiable” is a decidable statement.
• $\text{CL}$ has $n$ propositional variables, hence we have $2^n$ possible truth assignments $\nu$ to examine and evaluate whether $\nu(\text{CL}) = 1$ or $\nu(\text{CL}) = 0$
• This is called Semantical Decidability
• **Problem:** Exponential complexity
Syntactical Decidability Method: Resolution Deduction

• **Goal**: We want to show that a finite set $CL$ of clauses is **unsatisfiable**

• **Method**: Resolution deduction:

  • **Start** with $CL$; apply a transformation rule called **Resolution** as long as it is possible.

  • **If** you get $\emptyset$, then answer is **Yes**, i.e. $CL$ is **unsatisfiable**

  • **If** you **never** get $\emptyset$, then answer is **NO**, i.e. $CL$ is **satisfiable**
Resolution Completeness Theorem 1

Completeness of the Resolution:

CL is unsatisfiable iff we obtain the empty clause \{\} by a multiple use of the Resolution Rule

• Symbolically: \( \text{CL} \vdash \{\} \)

• It means we deduce the empty clause \{\} from \text{CL} by use of the resolution rule;

• We prove \{\} from \text{CL} by resolution
Resolution Completeness Theorem 1

|= CL denotes CL is a tautology

|=| CL denotes CL is unsatisfiable (contradiction)

• We write symbolically:

Resolution Completeness Theorem 1

|=| CL iff CL ⊢ {}
Refutation

• **Refutation:** proving the contradiction

In classical logic we have that:

A formula $A$ is a **tautology** iff $\neg A$ is a *contradiction*

Symbolically: $|= A$ iff $|= \neg A$

Observe:

$|= (A_1 \land \ldots \land A_n \Rightarrow B)$ iff $|= (A_1 \land \ldots \land A_n \land \neg B)$

Because $\neg (A \Rightarrow B) \equiv (A \land \neg B)$
Refutation

By **Resolution Completeness Theorem** this is **almost** equivalent to

\[ |= (A_1 \land \ldots \land A_n \Rightarrow B) \iff (A_1 \land \ldots \land A_n \land \neg B) \vdash \{ \} \]

**Almost**- means not YET Resolution works for **clauses** not formulas!

The **IDEA** is the following:

- to prove \( B \) from \( A_1, \ldots, A_n \) we keep \( A_1, \ldots, A_n \), ADD \( \neg B \) to it and use the **Resolution Rule**
- If we get \( \{ \} \), we have proved \( (A_1 \land \ldots \land A_n \Rightarrow B) \)

It is called a **proof by REFUTATION**; to prove \( C \) we start with \( \neg C \) and if we get a contradiction \( \{ \} \), we have proved \( C \)
Formulas – Clauses

Resolution works only for clauses

To use Resolution Deduction we need to transform our formulas into clauses i.e. we need to prove the following

Theorem

For any formula $A \in F$, there is a set of clauses $CL_A$ such that $A$ is logically equivalent to the set of clauses $CL_A$

$CL_A$ is called a clausal form of the formula $A$

We have good set of rules for automatic transformation of $A$ into its clausal form and we will study it as next step
Completeness

• **Resolution Completeness 2**
  For any propositional formula \( A \)
  
  \[ |= A \iff \text{CL}_{\neg A} \vdash \{\} \]

  where \( \text{CL}_{\neg A} \) is the clausal form of \( \neg A \)

• **Resolution Proof of A definition:**
  
  \[ \vdash_{R} A \iff \text{CL}_{\neg A} \vdash \{\} \]

**Resolution Completeness 2:**

\[ |= A \iff \vdash_{R} A \]
Resolution Rule  R

• \( C_1(a) \) means: clause \( C_1 \) contains a positive literal \( a \)

• \( C_2(\neg a) \) means: clause \( C_2 \) contains a negative literal \( \neg a \)

• **Resolution Rule  R** (two Premises)

  \[ C_1(a) : C_2(\neg a) \]

  Resolve on \( a \)

  \((C_1-\{a\} \cup C_2-\{\neg a\}) \leftarrow \text{Resolvent}\)

Clauses \( C_1(a) \) and \( C_2(\neg a) \) are called a **complementary pair**
Resolution Rule

- **Resolution Rule** takes 2 clauses and returns one. We usually write it in a form of a graph:

- **Definition:** \( C_1(a), C_1(\neg a) \) is called a complementary pair

- \( C_1(a) \quad C_1(\neg a) \)

\[ (C_1-\{a\}) \cup (C_2-\{\neg a\}) \leftarrow \text{Resolvent on } a \]
Resolution Rule R

• Clauses are SETS!

• \( \{C_1, C_2\} \) Complementary Pair

\[
C_1 = \{ a, b, c, \neg d \} \quad \quad \quad \quad C_2 = \{ \neg a, \neg b, d \}
\]

Resolve on \( a \)

\[
\{ b, c, \neg d, \neg b, d \} \quad \text{Resolvent on } a
\]
Example

C_1 = \{a, b, c, ¬d\} \quad C_2 = \{¬a, ¬b, d\}

\{ a, c, ¬d, ¬a, d\}

- Resolution Rule: R (Two Premises)

\[ \begin{align*}
C_1(b) : C_2(¬b) \quad \text{Resolve on } b \\
(C_1\{-b\} \cup C_2\{-¬b\}) \leftarrow \text{Resolvent}
\end{align*} \]
Exercise

- **CL** - set of clauses

**Find all resolvents of CL**

It means locate all clauses in **CL** that are Complementary Pairs and Resolve them

\[ C_1 = \{a, b, c, \neg d\} \quad C_2 = \{\neg a, \neg b, d\} \]

\[ CL = \{C_1, C_2\} \] has 3 Complementary Pairs

\[ C_1(a), C_2(\neg a) - P1 \]
\[ C_1(b), C_2(\neg b) - P2 \]
\[ C_2(d), C_1(\neg d) - P3 \]
Example

- \( CL = \{C_1, C_2\} = \{C_2, C_1\}\)

\( C_1 = \{a, b, c, \neg d\} \quad \text{and} \quad C_2 = \{\neg a, \neg b, d\}\)

Remember:

Resolution Rule uses **one literal** at the time!

\( C_1(a); C_2(\neg a) \text{ Resolve on } a \) : we get \( \{b, c, \neg d, \neg b, d\} \)

\( C_1(b); C_2(\neg b) \text{ Resolve on } b \) : we get \( \{a, c, \neg d, \neg a, d\} \)

\( C_1(d); C_2(\neg d) \text{ Resolve on } d \) : we get \( \{a, b, c, \neg a, \neg b\} \)
Example

\( C_1(b) : C_2(\neg b) \)  
Pair \( \{C_1 \ C_2\} \)

\((C_1-\{b\}) \cup (C_2-\{\neg b\})\)

\( \{a, \ b, \ c, \ \neg \ d\} \ \{\neg \ a, \neg \ b, \ d\} \)

Resolve on \( b \)

\( \{a, \ c, \ \neg d, \ \neg a, \ d\} \ \leftarrow \ Resolvent \ on \ b \)
Example

\( C_1(d) : C_2(\neg d) \) on \( \{C_1, C_2\} \)

\((C_1-\{d\}) \cup (C_2-\{\neg d\})\)

\{a, b, c, \neg d\} ; \{\neg a, \neg b, d\}

Resolve on d

\{a, b, c, \neg a, \neg b\}
Example

\[ C_1 = \{a, b, c, \neg d\} ; C_2 = \{\neg a, \neg b, c, d\} \]

Resolve on \(b\)

\[ \{a, c, \neg d, \neg a, d\} \]

Two clauses (one complementary pair) can have more than one resolvent – you can also resolve the complementary pair \(C_1 C_2\) on \(a\)
Example

• We can also resolve \( \{C_1 C_2\} \) on \( \{a, b, c, \neg d\} \) and \( \{\neg a, \neg b, d\} \) to resolve on \( a \):

\[ \{b, c, \neg d, \neg b, d\} \]

These are all resolvent of pair \( \{C_1 C_2\} \):

\[ \{b, c, \neg d, \neg b, d\}, \{a, c, \neg d, \neg a, d\} \]
\[ \{a, b, c, \neg a, \neg b\} \]
Resolution Deduction

- **CL** - set of clauses

**Procedure:** Deduce a clause \( C \) from **CL**: \( CL \vdash_{R} \{C\} \)

**Start** with **CL**, apply the resolution rule \( R \) to **CL**
Add resolvent to **CL** and
Repeat adding resolvents to already obtained set of resolvents until you get \( C \)

**Example**

**CL** = \{\{a, b\}, \{\neg a, c\}, \{\neg b, c\}\}

R on a \( \{b, c\} \)

R on b \( \{c\} \)

\( CL \vdash_{R} \{c\} \)
Example

- $\mathbf{CL} = \{\{a, b\}, \{-a, c\}, \{-b, c\}\}$

We have 2 possible deduction of $\{c\}$ from $\mathbf{CL}$

$$\mathbf{CL} \vdash R \{c\}$$
Example

• \( \text{CL} = \{\{a, b\}, \{-a, c\}, \{-b, c\}, \{-c\}\} \)

\( \{b, c\} \)

\( \{c\} \)

\( \{\} \)

\( \text{CL} \models_R \{\} \)

\( \text{CL is unsatisfiable} \) by Completeness Theorem

\( = |\text{CL} \) iff \( \text{CL} \models_R \{\} \)

Resolution deduction is not unique!

Next: Strategies for Resolution
Example

- \( \text{CL} = \{\{a, b\}, \{-a, c\}, \{-b, c\}, \{-c\}\} \)

Another deduction of \( \{\} \) from \( \text{CL} \)
Exercise

• Let $\text{CL} = \{\{a, b\}, \{-a, c\}, \{-b, c\}\}$

Find all possible deduction from $\text{CL}$

Remember:

1. If you get $\{\}$, it means $\text{CL}$ is unsatisfiable.
2. If you never get $\{\}$, it means $\text{CL}$ is satisfiable.

1 and 2 is true by Completeness Theorem:

$$=| \text{CL} \iff \text{CL} \vdash \{\}$$

$\text{CL}$ is unsatisfiable iff there is a deduction of $\{\}$ from it

$\text{CL}$ is satisfiable iff there is NO deduction of $\{\}$ from it
Exercise

\[ \text{CL} = \{\{a, b\}, \{-a, c\}, \{-b, c\}\} \]

Derivation 1: \{\{a, b\}, \{-a, c\}, \{-b, c\}\}

- R on a \{b, c\}
- \{c\} R on b STOP

Derivation 2: \{\{a, b\}, \{-a, c\}, \{-b, c\}\}

- R on b \{a, c\}
- \{c\} R on a STOP

No more (possible) Derivations, i.e. by Completeness Theorem we have that CL is satisfiable
Exercise

• **CL** is **unsatisfiable** iff there is deduction of {} from it, i.e.

\[\text{CL} \vdash_R \{\}\]

**CL** is **satisfiable** iff never \(\text{CL} \vdash_R \{\}\) (must cover all possibilities of deduction)

\[\text{CL} = \{\{a, b\}, \{\neg b\}, \{a, c\}, \{\neg a, d\}\}\]

\{a\}

\{b, d\}

\{c, d\}  \text{ STOP}

This is just **one** derivation.
You must consider **ALL possible** derivations and show that none ends with {} to prove that **CL** is **satisfiable**
Exercise

• Given: \( CL = \{C_1, C_2, C_3, C_4\} \)
  \[ CL = \{\{a, b, \neg b\}, \{-a, \neg b, d\}, \{a, b, \neg c\}, \{-a, c, b, e\}\} \]

1. Find all complementary pairs. Here they are:
   \( \{C_1, C_2\}, \{C_1, C_4\}, \)
   \( \{C_3, C_2\}, \{C_2, C_3\}, \)
   \( \{C_3, C_4\}, \{C_2, C_4\} \)

2. Find all resolvents for your complementary pairs
   For example: \( C_1 = \{a, b, \neg b\}, C_2 = \{-a, \neg b, d\} \) has 2 resolvents.
   Resolve on \( a \): \( \{-b, d, b\} \)
   Resolve on \( b \):
   \( \{a, \neg a, d, \neg b\} \)
Exercise

- \( CL = \{C_1, C_2\}, \) for \( C_1 = \{a, b, c, \neg d\}, \ C_2 = \{\neg a, \neg b, d\} \)

\( CL \) has 3 resolvents :-

1. \( \{\neg a, \neg b, a, b, c\} \) – resolve on d
2. \( \{\neg a, c, \neg d, d, a\} \) – resolve on b
3. \( \{b, c, \neg d, d\} \) – resolve on a

Let now \( CL = \{C_1, C_2, C_3\}, \) for \( C_1 = \{a\}, \ C_2 = \{b, \neg a\}, \ C_3 = \{\neg b, \neg a\} \)

Exercise:

Find all Complementary Pairs + find all their resolvents
Propositional Resolution
Part 2
GOAL: Use Resolution to prove/ disapprove  \( \models A \)

PROCEDURE

Step 1: Write \( \neg A \) and transform \( \neg A \) info set of clauses \( \text{CL}{\neg A} \) using Transformation rules

Step 2: Consider \( \text{CL}{\neg A} \) and look at if you can get a deduction of \( \{\} \) from \( \text{CL}{\neg A} \)

ANSWER

1. \( \text{CL}{\neg A} \models \text{R} \{\} \) — Yes, \( \models A \)

2. \( \text{CL}{\neg A} \vdash \{\} \) (i.e. you never get \( \{\} \)) — No, not \( \models A \)
Rules of transformation

• **Rules of transformation** of a formula $A$ into a logically equivalent set of clauses $\text{CL}_A$

• **Rule (U): (AUB) + Information**

What "Information" mean?

Example: $a, b, (a \cup \neg (a \Rightarrow b)), \neg c$

$a, b, a, \neg (a \Rightarrow b), \neg c$

$a,b, \neg c$ is Information

Rule (U): $I, (AUB), J$

$I, A, B, J$

$I,J$ --- Information around
Implication Rule (⇒)

- I, (A⇒B), J (A⇒B)

I, ¬A, B, J ¬A, B

Example: a, (a U b), (a ⇒ ¬a), (a ∧ b), c (⇒)

a, (a U b), ¬a, ¬a, (a ∧ b), c (U)
a, a, b, ¬a, ¬a, (a ∧ b), c

next step?
we need (∧) Rule!
Conjunction Rule ($\land$)

I, (A $\land$ B), J

I, A, J    I, B, J

(A $\land$ B)

A    B

Example:

a, a, b, ¬a, ¬a, (a $\land$ b), c

a, a, b, ¬a, ¬a, a, c    a, a, b, ¬a, ¬a, b, c

STOP when get only literals

Form clauses out of the leaves
Set of Clauses

Procedure: Leaves – to – Clauses

1. make SETS out of each leaf; each leaf becomes a clause C

2. make a set of clauses CL as a set of all clauses C obtained in 1.

   Leaf 1: \{a, a, b, \neg a, \neg a, a, c\} = \{a, b, \neg a, c\}
   Leaf 2: \{a, a, b, \neg a, \neg a, b, c\} = \{a, b, \neg a, c\}

• Observe that we end-up with only one set of clauses

• \[ CL = \{\text{Leaf 1, Leaf 2}\} = \{ \{a, b, \neg a, c\} \} \]
Negation of Implication Rule ($\neg = >$)

$I, \neg (A => B), I'$

$(\neg = >)$

$I, A, I', I, \neg B, I'$

$\neg (A => B)$

$(\neg = >)$

$A, \neg B$

Example:

$a, b, a, \neg (a => b), \neg c$

$(\neg = >)$

$a, b, a, \neg c$

$(\neg = >)$

$a, b, a, \neg b, \neg c$

Stop - when only literals:

Form clauses **out of** $a, b, a, a, \neg c$ and $a, b, a, \neg b, \neg c$
Clauses

• Leaf1: $a, b, a, a, \neg c$ makes clause $\{a, b, \neg c\}$
• Leaf 2: $a, b, a, \neg b, \neg c$ makes clause $\{a, b, \neg b, c\}$

• $CL = \{\{a, b, \neg c\}, \{a, b, \neg b, c\}\}$

• $CL$ is set of clauses corresponding to $a, b, a, \neg (a \Rightarrow b), \neg c$
Negation of Disjunction Rule ($\neg U$)

I, $\neg(A \cup B)$, J

$\neg U$

I, $\neg A$, J   I, $\neg B$, J

$\neg A$   $\neg B$

• Corresponds to DeMorgan Law:

$\neg(A \cup B) \equiv (\neg A \land \neg B)$
Negation of Conjunction Rule ($\neg (\land)$)

$I, \neg (A \land B), J$

$\neg (A \land B)$

$I, \neg A, \neg B, J$

$\neg A, \neg B$

Corresponds to DeMorgan Law

$\neg (A \land B) \equiv (\neg A \lor \neg B)$
Negation of Negation Rule ($\neg\neg$)

$I, \neg\neg (A), J$

$(\neg\neg)$

$I, A, J$

$\neg\neg (A)$

$(\neg\neg)$

$A$

Corresponds to

$\neg\neg (A) \equiv A$

Transformation Rules:

$(\wedge), (\cup), (\Rightarrow), (\neg \wedge), (\neg \cup), (\neg \Rightarrow)$
Transformation Rules Shorthand Form

- **(A ∨ B)** (U)
- **(¬A, ¬B)**

- **(A ∧ B)** (∧)
- **(¬A, ¬B)**

- **(A → B)** (⇒)
- **(¬A, B)**

- **¬¬A** (¬¬)
- **A**

- **¬(A ∨ B)** (¬U)
- **¬A**
- **¬B**

- **¬(A ∧ B)** (¬∧)
- **¬A, ¬B**

- **¬(A → B)** (¬⇒)
- **¬A, B**

+ Keep all Information

End when all leaves are literals
Example

• Let $A$ be a Formula $(((a \rightarrow \neg b) \cup c) \land (\neg a \cup \neg b))$
• Find $\text{CL}_A$

$(((a \rightarrow \neg b) \cup c) \land (\neg a \cup \neg b))$

$(a \rightarrow \neg b), c$

$\{\neg a, \neg b\} \text{ STOP}$

$\{\neg a, \neg b, c\} \text{ STOP}$

$\text{CL}_A = \{\{\neg a, \neg b, c\}, \{\neg a, b\}\}$

$A \equiv \text{CL}_A$
ARGUMENTS (rules of inference)

• From (premises) $A_1, \ldots, A_n$ we conclude $B$

$$A_1, \ldots, A_n \Rightarrow B$$

Definition:
Argument $A_1, \ldots, A_n$ is VALID iff

$$| = ((A_1 \land \ldots \land A_n) \Rightarrow B)$$
ARGUMENTS

• Otherwise
  Argument is NOT VALID

Valid Arguments ≡ Tautologically Valid

$A_1, \ldots, A_n, C$

are formulas of Propositional or Predicate Language
Validity of Arguments

Remember:  \( |= A \iff =| \neg A \)

Tautology (always true), Contradiction (always false)

This means that if we want to decide \( |= A \) we decide \( =| \neg A \)
and use Resolution for that

**STEPS**

**Step 1:** Negate \( A \); i.e. take \( \neg A \) and find the set of clauses corresponding to \( \neg A \) i.e. find \( \text{CL}\{\neg A\} \)

**Step 2:** Use Completeness of Resolution

\[ |= A \iff \text{CL} \{\neg A\} \vdash_R \{\} \]
i.e.

1. Look for a deduction of \( \{\} \)
2. if YES – we have \( |= A \)
3. If there is no deduction of \( \{\} \) we have: \( |= A \)
Basic Theorems

**T1.** \( = | \text{CL} \iff \text{CL} \vdash_R \{ \} \)

**CL** is inconsistent iff there is a resolution deduction of \( \{ \} \) from **CL**

**T2.** For any formula \( A \), there is a set of clauses \( \text{CL}_A \) such that \( A \equiv \text{CL}_A \)

**T3.** \( |= A \iff =| \neg A \)

By **T2** we get that

\( |= A \iff =| \text{CL}_{\neg A} \)

And by **T1** and **T3** we get

**T4.** \( |= A \iff \text{CL}_{\neg A} \vdash_R \{ \} \)
Exercise

• **Prove** By **Propositional Resolution**
• $|= (\neg(a=>b) \Rightarrow (a \land \neg b))$

**Remember:** $|= A$ iff $=| \neg A$ + use **Resolution**

**Steps**

**Step 1:** Find set of clauses corresponding to $\neg A$

i.e. $\text{CL} \{\neg A\}$

**Step 2:** Find deduction of $\{\}$ from $\text{CL} \{\neg A\}$

i.e. show that $\text{CL} \{\neg A\} \vdash R \{\}$

**DO IT!**
Exercise Solution

• **Step 1:** Negate A and find the set of clauses for ¬A i.e. \( \text{CL}_{\neg A} \)

• \( \neg (\neg (a \Rightarrow b) \Rightarrow (a \land \neg b)) \)

\[
\begin{align*}
\neg (a \Rightarrow b) & \quad \neg (a \land \neg b) \\
\neg (\neg (a \Rightarrow b)) & \quad \neg (a \land \neg b) \\
a & \quad \neg b & \quad \neg a, \neg \neg b \\
\{a\} & \quad \{\neg b\} & \quad \{\neg a, b\} \\
\text{CL }_{\neg A} & = \{\{a\}, \{\neg b\}, \{\neg a, b\}\}
\end{align*}
\]

**Step 2:** Check if \( \text{CL}_{\neg A} \vdash_R \{\} \) — YES!

\[
\{\} \quad \{\} \\
\]

**Remark:** \( \models A \) iff there is no deduction of \( \{\} \) from \( \text{CL}_{\neg A} \)
Back To Arguments

• Use resolution to show that from $A_1, \ldots, A_n$ we can deduce $B$

“We can” deduce $B$ from $A_1, \ldots, A_n$ means validity of argument $A_1, \ldots, A_n \Rightarrow B$

iff by definition

$\models (A_1 \land \ldots \land A_n \Rightarrow B)$

We have to use Resolution to prove that this is a Tautology
Arguments

\[ \models (A_1 \land \ldots \land A_n \Rightarrow B) \iff \neg \models (\neg (A_1 \land \ldots \land A_n \Rightarrow B)) \iff \models (A_1 \land \ldots \land A_n \land \neg B) \]

- **Step 1:** we transform \((A_1 \land \ldots \land A_n \land \neg B)\) to clauses
- Take \(A_1, \ldots, A_n\) and find \(\text{CL}_{A_1}, \ldots, \text{CL}_{A_n}\) and also find \(\text{CL}_{\neg B}\)
  and form
  \[ \text{CL}_{A_1} U \ldots \text{CL}_{A_n} U \text{CL}_{\neg B} = \text{CL} \]

**Step 2:** examine whether \(\text{CL} \vdash_R \{\}\)
Remember

- Argument $A_1, \ldots, A_n$ is valid iff $B$
  
  $\text{CL}_{A_1} \cup \ldots \cup \text{CL}_{A_n} \cup \text{CL}_{\neg B} \vdash_R \{}$

  Argument is not valid iff never $\text{CL}_{A_1} \cup \ldots \cup \text{CL}_{A_n} \cup \text{CL}_{\neg B} \vdash_R \{}$

We have some Resolution Strategies that allow us to cut down number of cases to consider
Example

Check if you can deduce

\[ B = \neg(a \cup \neg b) \Rightarrow (\neg a \land b) \]

from \( A_1 = ((a \Rightarrow \neg b) \Rightarrow a) \) and \( A_2 = (a \Rightarrow (b \Rightarrow a)) \)

Procedure:
1. Find \( \text{CL}_{\{A_1\}} \), \( \text{CL}_{\{A_2\}} \) and \( \text{CL}_{\{\neg B\}} \)
2. Form \( \text{CL} = \text{CL}_{\{A_1\}} \cup \text{CL}_{\{A_2\}} \cup \text{CL}_{\{\neg B\}} \)
3. Check if \( \text{CL} \not\vdash \emptyset \) or if never \( \text{CL} \not\vdash \emptyset \)

Yes, we can

No, we can’t
A1 = ((a => ¬b) => a)

We get: \( \text{CL}_{A1} = \{\{a\}, \{b, a\}\} \)

A2 = ((a => (b => a))

We get: \( \text{CL}_{A2} = \{\neg a, \neg b, a\} \)
Example Solution

- \( \neg B = \neg (\neg (a \cup \neg b) \Rightarrow (\neg a \land b)) \)

\( \neg (a \cup \neg b) \)

\( \neg a \quad \neg \neg b \)

\( b \quad \neg \neg a, \neg \neg b \quad a, \neg b \)

\( CL = \{\{a\}, \{b, a\}, \{\neg a, \neg b, a\}, \{\neg a\}, \{b\}, \{a, \neg b\}\} \)

Remove Tautology Strategy gives us the set

\( CL = \{\{a\}, \{b, a\}, \{\neg a\}, \{b\}, \{a, \neg b\}\} \)
Example Solution

- \( CL = \{\{a\}, \{b, a\}, \{\neg a, \neg b, c\}, \{\neg a\}, \{b\}, \{a, \neg b\}\} \)

\( \{a\} \)  R on \( b \)

\( \{\} \)

Yes  Argument is Valid

Next: Strategies for Resolution
Propositional Resolution
Part 3
Resolution Strategies

• We present here some Deletion Strategies and discuss their Completeness.

Deletion Strategies are restriction techniques in which clauses with specified properties are eliminated from set of clauses CL before they are used.
Pure Literals

Definition

A literal is **pure** in **CL** iff it has no complementary literal in any other clause in **CL**

Example: 

**CL** = \{ \{a, b\}, \{¬ c, d\}, \{c, b\}, \{¬ d\}\} 

a, b are **pure** and c, d, ¬c, ¬d are **not pure**

c has complement literal ¬c in \{¬c, d\} and 
¬c has complement literal c in \{c, b\}

d has a complement literal ¬d in the clause \{¬d\} and 
¬d has a complement literal d in \{¬c, d\}
S1: Pure Literals Deletion Strategy

S1 Strategy: **Remove all clauses that contain Pure Literals**

Clauses that contain pure literals are useless for retention process.

One pure literal in a clause is enough for the clause removal

This Strategy is complete, i.e.

\[ \text{CL} \vdash {} \iff \text{CL'} \vdash {} \]

where \( \text{CL}' \) is obtained from \( \text{CL} \) by pure literal clauses deletion
Example

- \( \textbf{CL} = \{\{\neg a, \neg b, c\}, \{\neg p, d\}, \{\neg b, d\}, \{a\}, \{b\}, \{\neg c\}\} \)

  \(d, \neg p\) are pure,

- \( \textbf{CL}' = \{\{\neg a, \neg b, c\}, \{a\}, \{b\}, \{\neg c\}\} \)

\(\{\neg b, c\}\)

\(\{c\}\)

\(\{\}\)
S2. Tautology Deletion Strategy

- **Tautology** — a clause containing a pair of complementary literals (a and \( \neg a \))

- **S2: Tautology Deletion:**
  \[ CL' = \text{Remove all Tautologies from } CL \]

- Example:
  \[ CL = \{ \{ a, b, \neg a \}, \{ b, \neg b, c \}, \{ a \} \} \]
  \[ CL' = \{ \{ a \} \} \]

- Tautology Deletion Strategy S2 is **COMPLETE**.

  \( CL \) is satisfiable \( \equiv \)
  \( CL' \) is satisfiable

  \( CL \) unsatisfiable \( \equiv \)
  \( CL' \) unsatisfiable
Exercise

Example:

- Example: CL = {{ a, ¬a, b}, {b, ¬b, c}} - remove tautologies; CL’ has no elements, i.e. CL’ = φ

CL is always satisfiable and so is CL’ as Φ is always satisfiable!

Exercise

Prove correctness of Tautology Deletion Strategy
S3. Unit Resolution Strategy

- **A unit resolvent** – resolvent in which at least one of the parent clauses is a unit clause i.e. is a clause containing a single literal.

- **A unit deduction** – all derived clauses are unit resolvents.

- **A unit Refutation** – unit deduction of the empty clause { }.

- **Example:** \{\{a, b\}, \{-a, c\}, \{-b, c\}, \{-c\}\}

  \begin{align*}
  &\{-a\} & \{-b\} \\
  &\{b\} \\
  &\{ \} \quad \text{Efficient but not Complete!}
  \end{align*}
Unit Resolution not complete

Example

- \( \text{CL} = \{\{a, b\}, \{-a, b\}, \{a, \neg b\}, \{-a, \neg b\}\} \)

\( \{b\} \)

\( \{a\} \)

\( \{\neg a\} \)

\( \{\}\)  

**CL is unsatisfiable**, but does not have unit deduction.

**Horn Clause:** a clause with at most one positive literal.

**Theorem:** Unit Resolution is complete on Horn Clauses.
**Example of Unit Resolution Deduction**

- **$CL = \{\{-a, c\}, \{-c\}, \{a, b\}, \{-b, c\}, \{-c\}\}$$

- $\{-a\}$
- $\{b\}$
- $\{c\}$
- $\{}$

$CL$ is not Horn but $CL \vdash \{}$ by unit deduction.

Remark: if we get $\{}$ by unit deduction we are OK but if we don’t get $\{}$ by unit deduction it does not mean that $CL$ is satisfiable, because unit strategy is not a Complete Strategy on non-Horn clauses.
S4. Input Resolution

- **Input Resolution** - At least one of the two parent clauses is in the initial database.
- **Input Deduction** - all derived clauses are input resolvents
- **Input Refutation** - Input deduction of \{
  
**THM 1:** Unit and Input Resolution are equivalent.

**THM 2:** Input Resolution is complete only on Horn Clauses
Example: $\text{CL} = \{\{a, b\}, \{\neg a, c\}, \{\neg b, c\}, \{\neg c\}\}$

NOT Complete!
5. Linear Resolution

- **Linear Resolution** also called Ancestry-Filtered resolution is a slight generalization of Input Resolution.

- **A Linear Resolution**: At least one of the parents is either in the initial DB or is in an Ancestor of the other parent.

- **A Linear Deduction**: Uses only linear resolvents: each derived clauses is a linear resolvent

- **A Linear Refutation**: Linear deduction of \( \{ \} \).

- Linear Resolution is complete
Example

$$CL = \{\{a, b\}, \{-a, b\}, \{a, -b\}, \{-a, -b\}\}$$

Here:

\{a\} is a parent of \{-b\}

\{b\} is the ancestor of \{-b\} (other parent of \{-b\})
Linear Resolution

Linear Resolution is complete
There are also more modifications of the LR that are complete

Our Strategies work also for **Predicate Logic Resolution**
First papers

Kowalski 1974, 1976 “Logic for problem solving” “Predicate Logic as a programming language”.

Robinson 1965 “A Machinery Oriented logic based on the resolution principle” J Assoc. for Computing Machinery 12(1)