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: \( \neg a, \neg b, \neg d \) ...

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

**Clause** – any finite set of literals

**Example:** \( C_1, C_2, C_3 \) are clauses where

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

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

Finite set of clauses

\[ \text{CL} = \{ C_1, \ldots, C_n \} \]

Example

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

- Think semantically of a clause
- \( C = \{ a_1, \ldots, a_n \} \) as disjunction, i.e. \( C \) is logically equivalent to
  \[ a_1 \lor a_2 \lor \ldots \lor a_n \quad \text{where} \quad a_i \in \text{Literal} \]
- 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
  \[ 0 – \text{False}, \quad 1 – \text{True} \]
  Shorthand : \( v^* = v \)
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:

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

\text{OBSERVE} that $v(C) = 1$ for all $v$, i.e. the clause

$C = \{ a, \neg b, c, \neg a \}$ is a \textbf{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 \( \text{a model for } C \)

3. A clause \( C \) is \textit{satisfiable} iff it has a \textit{model}, i.e.

\( C \) is \textit{satisfiable} iff there is a \( v \) such that \( v \models C \)

4. A clause \( C \) is \textit{a tautology} iff \( v \models C \) for all \( v \), i.e. all truth assignments \( v \) are \textit{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-sequence (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 $\{\}$ is the only unsatisfiable clause

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

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

   $$v(\text{CL}) = v(C_1) \land \ldots \land v(C_n)$$

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 $CL$ is **satisfiable** iff it **has a model**, i.e. iff $\exists v \ v(\ CL) = 1$

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

Remark:

If $\{\} \in CL$ then $CL$ is unsatisfiable
Unsatisfiability

Consider a set of clauses

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

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

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

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

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 a v, such that \( v(\text{CL}) = 1 \)

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

- \( \text{CL} = \{ C_1, C_2, \ldots, C_n \} \) - syntax
- \( \text{CL} = C_1 \land \ldots \land C_n \) - 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 \(v\) to examine and evaluate whether \(v(\text{CL}) = 1\) or \(v(\text{CL}) = 0\)
• 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: CL ⊢ {}  
• It means we deduce the empty clause {} from CL by use of the resolution rule;
• We prove {} from 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

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

**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** \( \emptyset \), **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 \( \emptyset \), **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 $A$
  \[\models A \iff \text{CL}_{\neg A} \vdash \{\}\]
  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:**
\[\models A \iff \vdash_R A\]
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 $a$

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

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

  Resolve on 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

\begin{align*}
C_1 &= \{a, b, c, \neg d\} \\
C_2 &= \{\neg a, \neg b, d\}
\end{align*}

Resolve on a

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

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

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

- Resolution Rule: \( R \) (Two Premises)

\[
\underbrace{C_1(b) : C_2(\neg b)} \quad \text{Resolve on } b
\]

\[
(C_1 - \{b\} \cup C_2 - \{\neg b\}) \leftarrow \text{Resolvent}
\]
Exercise

- **CL** - set of clauses

Find all resolvents of **CL**

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

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

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

\[
C_1(a), C_2(\neg a) \quad P1
\]

\[
C_1(b), C_2(\neg b) \quad P2
\]

\[
C_2(d), C_1(\neg d) \quad P3
\]
Example

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

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

Remember:

Resolution Rule uses one literal at the time!

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

C₁(b) : C₂(¬b) Pair \{C₁, C₂\}

(C₁-{b}) U (C₂-{¬b})

{a, b, c, ¬d} \{¬a, ¬b, d\}

Resolve on b

{a, c, ¬d, ¬a, d} <- 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
\{a, b, c, \neg d\}, \{\neg a, \neg b, d\}

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** \(b\) \(\{c\}\)
    - **R on** \(b\) \(\{c\}\)
      - **Add** resolvent to \(CL\): \(CL \vdash_R \{c\}\)
Example

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

We have 2 possible deduction of \( \{c\} \) from \( \text{CL} \)

\( \text{CL} \vdash_R \{c\} \)
Example

• CL = {{ a, b}, {¬ a, c}, {¬ b, c}, {¬ c}}
   {b,c}
   {c}
   {}

CL ⊢ \{\}

CL is unsatisfiable by Completeness Theorem

= |CL iff CL ⊢ \{\}

Resolution deduction is not unique!

Next: Strategies for Resolution
Example

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

Another deduction of $\{\}$ from CL
Exercise

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

Find all possible deduction from \( CL \)

Remember:
1. If you get \( \{ \} \), it means \( CL \) is unsatisfiable.
2. If you never get \( \{ \} \), it means \( CL \) is satisfiable.

1 and 2 is true by Completeness Theorem:

\[
= | \quad CL \quad \text{iff} \quad CL \vdash \{ \}
\]

\( CL \) is unsatisfiable iff there is a deduction of \( \{ \} \) from \( CL \)

\( CL \) is satisfiable iff there is NO deduction of \( \{ \} \) from \( CL \)
Exercise

- **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)

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

\[ \{a\} \]

\[ \{b, d\} \]

\[ \{d\} \text{ STOP} \]

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: \( CL = \{C_1, C_2, C_3, C_4\} \)

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

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: \( C_1 = \{a, b, \neg b\} , \ C_2 = \{\neg a, \neg b, d\} \) has 2 resolvents.
Resolve on \( a: \ \{\neg b, d, b\} \)
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 \( \emptyset \) from \( \text{CL}\{\neg A\} \)

ANSWER

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

2. \( \text{CL}\{\neg A\} \vdash \emptyset \) (i.e. you never get \( \emptyset \)) — 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):** $(A \cup B) + \text{Information}$

What "Information" mean?

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

$\begin{align*}
a, b, a, \neg( a \rightarrow b), \neg c
\end{align*}$

$a, b$ and $\neg c$ is Information

**Rule (U):** $I, (A \cup B), J$

$I, A, B, J$

$I, J \quad \text{--- Information around}$
Implication Rule ($\Rightarrow$)

- I, $(A\Rightarrow B)$, J
  - I, $\neg A$, B, J

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

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

\[ I, (A \land B), J \]

\[ (\land) \]

\[ I, A, J \]
\[ I, B, J \]

\[ (A \land B) \]

\[ (\land) \]

\[ A \]
\[ B \]

Example:

\[ a, a, b, \neg a, \neg a, (a \land b), c \]

\[ (\land) \]

\[ a, a, b, \neg a, \neg a, a, c \]
\[ a, a, b, \neg a, \neg a, b, c \]

**STOP** when get only literals – called leaves

**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

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

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

Example:

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

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

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

Stop – when only literals:
Form clauses out of leaves $a, b, a, a, \neg c$ 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 $\{a, b, \neg b, c\}$

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

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

I, ¬(A UB), J

¬(A UB)

I, ¬A, J

I, ¬B, J

¬A

¬B

Rule (¬ U) corresponds to DeMorgan Law:

¬(A UB) ≡ (¬A ∧ ¬B)
Negation of Conjunction Rule \((\neg \land)\)

\[
\begin{align*}
\text{I, } \neg(A \land B), & \quad \neg(A \land B) \\
\text{J, } (\neg \land) & \quad (\neg \land) \\
\text{I, } \neg A, \neg B, & \quad \neg A, \neg B \\
\text{J, } (\neg \land) & \quad (\neg \land)
\end{align*}
\]

Rule \((\neg \land)\) corresponds to DeMorgan Law

\[
\neg(A \land B) \equiv (\neg A \cup \neg B)
\]
Negation Rule (¬¬)

I, ¬¬A, J

(¬¬)

I, A, J

¬¬A

(¬¬)

A

Negation Rule (¬¬) Corresponds to

¬¬A ≡ A

Transformation Rules :

(∧), (U), (⇒), (¬∧), (¬U), (¬⇒)
Transformation Rules Shorthand Form

\[(A \cup B)\] (\(U\))
\[A, B\]

\[(A \land B)\] (\(\land\))
\[A \quad B\]

\[(A \Rightarrow B)\] (\(\Rightarrow\))
\[\neg A, B\]

\[\neg
\neg A\] (\(\neg\neg\))
\[A\]

End when all leaves are literals

\[\neg (A \cup B)\] (\(\neg U\))
\[\neg A \quad \neg B\]

\[\neg (A \land B)\] (\(\neg \land\))
\[\neg A, \neg B\]

\[\neg (A \Rightarrow B)\] (\(\neg \Rightarrow\))
\[A \quad \neg B\]

+ Keep all Information
Example

Let $A$ be a Formula $(((a \Rightarrow \neg b) \lor c) \land (\neg a \lor \neg b))$

Find $\text{CL}_A$

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

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

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

$\neg a, \neg b$ STOP

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

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

$A \equiv \text{CL}_A$
ARGUMENTS

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

  \[
  \overline{A_1, \ldots, A_n} \quad B
  \]

Definition:

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

\[
| = ((A_1 \land \ldots \land A_n) \Rightarrow B)
\]

• Otherwise Argument is NOT VALID
ARGUMENTS

Valid Arguments $\equiv$ Tautologically Valid

$A_1, \ldots, A_n, C$

can be 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 to do 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 resolution deduction of \( \{\} \) from \( \text{CL}\{\neg A\} \)
2. if YES – we have \( |= A \)
3. If there is no deduction of \( \{\} \) we have: NOT \( |= A \)
Basic Theorems

T1. \( | \text{CL} | \) iff \( \text{CL} \vdash_{\text{R}} \{ \} \)

**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 \( | \neg A \)

And by T1 and T3 we get

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

• Prove By Propositional Resolution

\[ \models (\neg (a \implies b) \implies (a \land \neg b)) \]

Remember: \[ \models A \iff \models \neg A \] + use Resolution

Steps

Step 1: Find set of clauses corresponding to \neg A i.e.
find \[ CL_{\neg A} \]

Step 2: Find deduction of \{\} from \[ CL_{\neg A} \]
i.e. show that \[ CL_{\neg A} \vdash_R \{\} \]

DO IT!
Exercise Solution

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

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

\[
\neg (a \Rightarrow b) \quad \neg (a \land \neg b)
\]

$\neg (a \Rightarrow b)$

$a$ \quad $\neg b$

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

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

$\neg a$, $b$

Clauses: $\{a\} \quad \{\neg b\} \quad \{\neg a, b\}$

$\text{CL}_{\neg A} = \{\{a\}, \{\neg b\}, \{\neg a, b\}\}$

Step 2: Check if $\text{CL}_{\neg A} \vdash \{\}$ – **YES!**

Remark: \(\not |= A\) iff there is **no** deduction of $\{\}$ from $\text{CL}_{\neg A}$
• 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 the argument $\frac{A_1, \ldots, A}{B}$

This means that we have to show that

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

We have to use **Resolution** to prove that $(A_1 \land \ldots \land A_n \Rightarrow B)$ is a **tautology**
Arguments

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

- **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 then **form**

  \[ \text{CL}_{A_1} \cup \ldots \cup \text{CL}_{A_n} \cup \text{CL}_{\neg B} = \text{CL} \]

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

Argument $A_1, \ldots , A_n$ is valid $B$

iff $\text{CL}_{A_1} \cup \ldots \cup \text{CL}_{A_n} \cup \text{CL}_{\neg B} \vdash R \{\}$

$\downarrow$

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 \( A1 = ((a \Rightarrow \neg b) \Rightarrow a) \) and \( A2 = (a \Rightarrow (b \Rightarrow a)) \)

Procedure:

1. Find \( CL\{A1\} \), \( CL\{A2\} \) and \( CL\{\neg B\} \)
2. Form \( CL = CL\{A1\} \cup CL\{A2\} \cup CL\{\neg B\} \)
3. Check if \( CL \not\vdash \{\} \) or if never \( CL \not\vdash \{\} \)

Yes, we can \hspace{5cm} No, we can’t
Example Solution

A1 = ((a => ¬b) => a)

We get: $CL_{A1} = \{\{a\}, \{b, a\}\}$

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

We get: $CL_{A2} = \{¬a, ¬b, a\}$
Example Solution

\( \neg B = \neg(\neg(a \cup \neg b) => (\neg a \wedge b)) \)

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

\( \neg (\neg a \wedge b) \)

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

\( a, \neg b \)

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

Remove Tautology Strategy gives us the set

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

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

\[ \begin{align*}
\{a\} & \xrightarrow{\text{R on } b} \{\} \\
\{\} &
\end{align*} \]

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: \( CL = \{\{a, b\}, \{\neg c, d\}, \{c, b\}, \{\neg d\}\} \)
a, b are pure and c, d, \neg c, \neg d are not pure.

c has complement literal \(\neg c\) in \(\{\neg c, d\}\) and
\(\neg c\) has complement literal \(c\) in \(\{c, b\}\)
d has a complement literal \(\neg d\) in the clause \(\{\neg d\}\) and
\(\neg d\) has a complement literal \(d\) in \(\{\neg c, d\}\)
**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.

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

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

- $\text{CL} = \{\{\neg a, \neg b, c\}\}, \{\neg p, d\}, \{\neg b, d\}, \{a\}, \{b\}, \{\neg c\}\}$
  
  $d, \neg p$ are pure,

$\text{CL}' = \{\{\neg a, \neg b, c\}\}, \{a\}, \{b\}, \{\neg c\}\}$
S2. Tautology Deletion Strategy

- **Tautology** – a clause containing a pair of complementary literals (a and \( \neg a \))
- **S2: Tautology Deletion:**
  - CL’ = 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:

• $\text{CL} = \{\{a, \neg a, b\}, \{b, \neg b, c\}\}$ - remove tautologies- get $\text{CL'}$ with no elements, i.e. $\text{CL'} = \emptyset$

$\text{CL}$ is always satisfiable and so is $\text{CL'}$ as $\emptyset$ 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\}, \{\neg a, c\}, \{\neg b, c\}, \{\neg c\}\} \\
  \{\neg a\} \quad \{\neg b\} \\
  \{b\} \\
  {} \quad \text{Efficient but not Complete!}
Unit Resolution not complete

Example

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

- \{b\}

- \{a\} \quad \{-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

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

$CL$ 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 \{\} is complete only on Horn Clauses

**THM 1**: Unit and Input Resolution are equivalent.

**THM 2**: Input Resolution is complete only on Horn Clauses
Input Resolution Deduction

Example: \[ \text{CL} = \{\{a, b\}, \{¬a, c\}, \{¬b, c\}, \{¬c\}\} \]
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

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

Here:

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

\{b\} is the ancestor of \{\neg b\} (other parent of \{\neg 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)