# Propositional Resolution Introduction

(Nilsson Book Handout)

Professor Anita Wasilewska
CSE 537 Artificial Intelligence

# Propositional Resolution
# Part 1

# SYNTAX “dictionary”

**Literal** – any **propositional** VARIABLE a or negation of a variable ¬ a, for a ∈ VAR

**Example:** variables: a, b, c .... negation of variables: ¬a, ¬b, -d ...

**Positive Literal:** any variable a ∈ VAR

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

**Example:** C1, C2, C3 are clauses where

C1 = {a, b} , C2 = {a, ¬ c} , C3 = { a, ¬a, ......,$a_k$ }

# Syntax "Dictionary"

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

**Finite set of clauses**

**CL** = { C1, …., Cn}

**Example**

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

# Semantics – Interpretation of Clauses

- Think **semantically** of a clause
- $C = \{ a_1, \ldots\ldots, a_n \}$ as disjunction, i.e.

  C is logically equivalent to

  $a_1 \cup a_2 \cup \ldots\ldots \cup a_n \qquad a_i \in$ Literal
- **Formally** – given a truth assignment $v : VAR \rightarrow \{0, 1\}$ we extended it to set of all CLAUSES **CL** as follows:

$$v^* : \quad \mathbf{CL} \rightarrow \{0, 1\}$$

$$v^*(C) = v^*(a_1) \cup \ldots\ldots \cup v^*(a_n)$$

for any clause C in **CL**, where

0 – False, 1 – True

**Shorthand** : $v^* = v$

# Satisfability, Model, Tautology

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

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

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

  We evaluate :

$v(C) = v(a) \cup \neg v(b) \cup v(c) \cup \neg v(a) =$

$1 \cup 0 \cup 0 \cup 1 = 1$

**OBSERVE** that $v(C) =1$ for all $v$, i.e. the clause

$C = \{ a, \neg b, c, \neg a\}$ is a **Tautology**

# Satisfability, Model, Tautology

Definitions

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

**2.** Any v such that **v l= C is called a MODE L 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 l= C**

**4.** A clause C is a **tautology** iff **v l= 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 ≠ 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 {} is the only unsatisfiable clause**

Let **CL** = { $C_1$, ….,$C_n$) be a **finite set of clauses**.

**2. We extended v : VAR -> {0, 1} to any set of clauses CL**

$$v(CL) = v(C_1) \wedge \ldots\ldots \wedge V(C_n)$$

**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 $\exists v \; v(\mathbf{CL}) = 1$

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

**Remark:**

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

# Unsatisfability

Consider a set of clauses

**CL** = {{a}, {a,b}, {¬ b}}

**CL** is **satisfiable** because any v, such that

v(a) =1, v(b) =0 is a **model** for **CL**

Check: v(CL) = 1 ∧ (1 U 0) ∧ 1 = 1

**FACT:** When {a} and {¬ a} are in **CL,**

then the set **CL** is **unstisfiable**

Remember: (a ∧ ¬ a ) is a contradiction

# Syntax and Semantics

- Example:
- C1 = { a, b, ¬c}, C2 ={c , a} - syntax
- C1 = a U b U ¬c - semantics
- C2 = c U a - semantics

- **CL** = {C1, C2} = {{a , b, ¬c} , {c , a}} – syntax

**CL** = (a U b U ¬c) ∧ (c U 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} - synatx
- **CL** = C1 ⋀ ...... ⋀ Cn - semantics

# Semantical Decidability

- A statement:
- " A finite set **CL** of clauses is/ is not satisfiable"

  is a **decidable statement.**
- **CL** has n propositional variables, hence we have $2 \wedge n$ possible truth assignments v to examine and evaluate whether **v(CL) = 1 or v(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 {},** then answer is **Yes**, i.e.

  **CL** is **unsatisfiable**
- **If** you **never get {}** , 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** $\vdash$ **{}**
- 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**

**$\models$ CL denotes CL is a tautology**

**$=\!|$ CL denotes CL is unsatisfiable** (contradiction)

- We write symbolically:

**Resolution Completeness Theorem 1**

$=\!|$ **CL** **iff** **CL** $\vdash$ **{}**

# Refutation

- **Refutation:** **proving the contradiction**

In classical logic we have that:

A **formula A is a tautology iff ¬A is a contradiction**

**Symbolically:** $\models A$ **iff** $=\mid \neg A$

**Observe:**

$\models (A1 \wedge \ldots\ldots \wedge An \Rightarrow B)$ **iff** $=\mid (A1 \wedge \ldots\ldots \wedge An \wedge \neg B)$

Because $\neg (A \Rightarrow B) \equiv (A \wedge \neg B)$

# Refutation

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

**$\models (A1 \wedge \ldots\ldots \wedge An \Rightarrow B)$ iff $(A1 \wedge \ldots\ldots \wedge An \wedge \neg B) \vdash \{\}$**

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

The **IDEA** is the following:

**to prove B from A1, ….,An we keep A1,….., An , ADD ¬B to it and use the Resolution Rule**

**If we get {}, we have proved $(A1 \wedge \ldots\ldots \wedge An \Rightarrow B)$**

**It is called a proof by REFUTATION; to prove C we start with ¬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** $\mathbf{CL}_A$ such that $A$ is logically equivalent to the set of clauses $\mathbf{CL}_A$

$\mathbf{CL}_A$ 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 $A$**

  $$\models A \quad \textbf{iff} \quad \mathbf{CL}_{\neg A} \vdash \{\}$$

  where $\mathbf{CL}_{\neg A}$ is the clausal form of $\neg A$

- **Resolution Proof of A definition:**

  $$\vdash_R A \quad \textbf{iff} \quad \mathbf{CL}_{\neg A} \vdash \{\}$$

**Resolution Completeness 2:**

$$\models A \quad \textbf{iff} \quad \vdash_R A$$

# 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)

$$\frac{C_1(a) : C_2(\neg a)}{(C_1 - \{a\} \cup C_2 - \{\neg a\})} \quad \text{Resolve on } a$$

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

Resolve on a

$(C_1-\{a\}) \cup (C_2-\{\neg a\})$ **<-** Resolvent on a

# Resolution Rule R

- Clauses are SETS!
- $\{C_1, C_2\}$ Complementary Pair

$C_1 = \{a, b, c, \neg d\}$ $\qquad$ $C_2 = \{\neg a, \neg b, d\}$

Resolve

on a

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

# Example

$C_1 = \{a ,b ,c , \neg d\}$ $\qquad$ $C_2 = \{\neg a ,\neg b ,d\}$

$\{b ,c ,d\}$

- **Resolution Rule: R (Two Premises)**
  **$\underline{C_1(a) : C_2(\neg a)}$** $\qquad$ **Resolve on a**
  **$(C_1-\{a\} \cup C_2-\{\neg a\})$ <- 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

$C_1 = \{a, b, c, \neg d\}$ $C_2 = \{\neg a, \neg b, d\}$

**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** $=\{C_1, C_2\} = \{C_2, C_1\}$

$C_1=\{a, b, c, \neg d\}$ $C_2=\{\neg a, \neg b, d\}$

**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

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

$(C_1 - \{b\})\ U\ (C_2 - \{\neg b\})$

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

Resolve on b

$\{a, c, \neg d, \neg a, d\}$ <- Resolvent on b

# Example

$\underline{C_1(d) : C_2(\neg d)}$ 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

$\{a, b, c, \neg d\}$ $\{\neg a, \neg b, d\}$ $\{C_1\ C_2\}$

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 o fresolvents

**until** you get C

**Example**

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

R on a {b,c}

R on b

{c }

**CL** $\vdash_R$ {c }

# Example

- **CL** = {{ a, b},{ ¬ a, c}, { ¬ b, c}}

Resolve on b

{a. c}

Resolve on a

{c}

We have 2 possible **deduction** of { c } from **CL**

$\mathbf{CL} \vdash_R \{ c \}$

# Example

- **CL** = {{ a, b},{ ¬ a, c},{ ¬ b, c}, {¬ c}}

{b,c}

{c}

{}  **CL** $\vdash_R$ {}

**CL is unsatisfiable** by **Completeness Theorem**

$\models$ **CL** **iff** **CL** $\vdash_R$ {}

Resolution deduction is not unique!

**Next:** Strategies for Resolution

# Example

- **CL**= {{ a, b},{ ¬ a, c},{ ¬ b, c}, {¬ c}}

{¬a}

{b}

{c}

{}

**Another deduction** of {} from **CL**

# Exercise

- Let **CL** = {{ a, b},{ ¬ a, c},{ ¬ 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:**

$\models$ **CL** iff **CL** $\vdash$ {}

**CL** is unsatisfiable **iff** there is a deduction of {} from it

**CL** is satisfiable **iff** there is NO deduction of {} from it

# 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.

  **CL** $\vdash_R$ {}

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

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

{a}

{b, d}

{c,d} 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 $\mathbf{CL}_{\{\neg A\}}$ using Transformation rules

**Step 2:** Consider $\mathbf{CL}_{\{\neg A\}}$ and look at if you can get a deduction of $\{\}$ from $\mathbf{CL}_{\{\neg A\}}$

ANSWER

1. $\mathbf{CL}_{\{\neg A\}} \vdash_R \{\}$ – Yes, $\models A$
2. $\mathbf{CL}_{\{\neg A\}} \nvdash \{\}$ (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 **CL**$_A$
- **Rule (U): (AUB) + Information**

What "Information" mean?

**Example:** a, b, (a U ¬( a=> b)), ¬c

a, b, a ,¬( a=> b), ¬c

a,b, ¬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 ($\wedge$)

**I, (A $\wedge$ B), J** ($\wedge$) **I , A, J**  **I , B, J**

**(A $\wedge$ B)** ($\wedge$) **A**  **B**

**Example:**

a, a, b, ¬a, ¬a, (a $\wedge$ b), c

($\wedge$)

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, ¬a, ¬a, a, c} = {a, b, ¬a, c}

Leaf 2: {a, a, b, ¬a, ¬a, b, c} = {a, b, ¬a, c}

- Observe that we end-up with only **one set** of clauses
- **CL** ={Leaf 1, Leaf 2} = { {a, b, ¬a, c} }

# Negation of Implication Rule (¬ =>)

I, ¬ (A =>B), I                                        ¬ (A => B)

(¬=>)                                                  (¬=>)

I , A, I     I , ¬B, I                                 A          ¬B

Example:

a, b, a, ¬ (a => b), ¬ c

(¬=>)

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

**Stop – when** only literals :

**Form clauses out of** a, b, a, a, ¬c    and

a, b, a, ¬b, ¬c

# Clauses

- Leaf1: a, b, a, a, ¬c makes clause {a, b, ¬c}
- Leaf 2: a, b, a, ¬b, ¬c makes clause {a, b, ¬b, c}

- **CL** = {{a, b, ¬c}, {a, b, ¬b, c}}

- **CL** is set of clauses corresponding to

  a, b, a, ¬ (a => b), ¬ c

# Negation of Disjunction Rule (¬ U)

I, ¬(A UB), J

(¬U)

I , ¬A, J       I , ¬B, J

¬(A U B)

(¬U)

¬A       ¬B

- Coresponds to DeMorgan Law:

  ¬(AUB) ≡ (¬A ⋀ ¬B)

# Negation of Conjunction Rule (¬∧)

$$\frac{I, \neg(A \wedge B), J}{I, \neg A, \neg B, J}\ (\neg\wedge)$$

$$\frac{\neg(A \wedge B)}{\neg A, \neg B}\ (\neg\wedge)$$

**Coresponds to DeMorgan Law**

$\neg(A \wedge B) \equiv (\neg A \cup \neg B)$

# Negation of Negation Rule (¬¬)

**I, ¬¬ (A), J**

**(¬¬)**

**I , A, J**

**¬¬(A)**

**(¬¬)**

**A**

**Coresponds to**

¬¬ (A) ≡ A

**Transformation Rules :**

( ⋀ ), (U), (=>), (¬ ⋀ ), (¬U), (¬=>)

# Transformation Rules Shorthand Form

**(AUB)** **(U)**

**A, B**

**¬(A U B)** **(¬U)**

**¬A** **¬B**

**(A ∧ B)** **(∧)**

**A** **B**

**¬(A ∧ B)** **(¬∧)**

**¬A, ¬B**

**(A=>B)** **(=>)**

**¬A, B**

**¬(A => B)** **(¬=>)**

**A** **¬B**

**¬¬A** **(¬¬)**

**A**

**+ Keep all Information**

**End** when all leaves are literals

# Example

- Let A be a Formula $(((a \Rightarrow \neg b) U c) \wedge (\neg a U \neg b))$
- **Find** $CL_A$
- $(((a \Rightarrow \neg b) U c) \wedge (\neg a U \neg b))$

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

$(a \Rightarrow \neg b), c$ $\{\neg a, b\}$ **STOP**

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

$$CL_A = \{\{\neg a, \neg b, c\}, \{\neg a, b\}\}$$

$$A \equiv CL_A$$

# ARGUMENTS (rules of inference)

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

$$\frac{A_1, \ldots\ldots, A_n}{B}$$

**Definition:**

Argument $\frac{A_1, \ldots\ldots, A_n}{B}$ is **VALID** **iff**

$$\models ((A1 \wedge \ldots\ldots \wedge An) \Rightarrow B)$$

# ARGUMENTS

- Otherwise

  Argument is **NOT VALID**

Valid Arguments ≡ Tautologically Valid

$A_1, \ldots\ldots, A_n, C$

are formulas of Propositional or Predicate Language

# Validity of Arguments

**Remember:** $\models A$ **iff** $=\!| \neg A$

Tautology (always true), Contradiction (always false)

This means that if we want to **decide** $\models A$ 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** $\mathbf{CL}_{\{\neg A\}}$

**Step 2:** Use Completeness of Resolution

$\models A$ **iff** $\mathbf{CL}_{\{\neg A\}} \vdash_R \{\}$ i.e.

1. Look for a deduction of $\{\}$
2. if YES – we have $\models A$
3. If there is no deduction of $\{\}$ we have: $\models A$

# Basic Theorems

**T1.** $=\mid$ **CL** **iff** **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 $\mathbf{CL}_A$ such that $A \equiv \mathbf{CL}_A$

**T3.** $\models A$ **iff** $=\mid \neg A$

By **T2** we get that

$\models A$ **iff** $=\mid \mathbf{CL}_{\{\neg A\}}$

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

**T4.** $\models A$ **iff** $\mathbf{CL}_{\{\neg A\}} \vdash_R \{\}$

# Exercise

- **Prove** By Propositional Resolution
- $\models (\neg(a \Rightarrow b) \Rightarrow (a \wedge \neg b))$

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

**Steps**

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

i.e. $\mathbf{CL}_{\{\neg A\}}$

**Step 2:** Find deduction of $\{\}$ from . $\mathbf{CL}_{\{\neg A\}}$

i.e. show that $\mathbf{CL}_{\{\neg A\}} \vdash_R \{\}$

**DO IT!**

# Exercise Solution

- **Step 1:** Negate A and find the set of clauses for ¬A i.e. $\mathbf{CL}_{\{\neg A\}}$
- $\neg(\neg(a \Rightarrow b) \Rightarrow (a \land \neg b))$

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

a $\qquad$ ¬b $\qquad\qquad$ ¬a, ¬¬b

{a} $\qquad$ {¬b} $\qquad\qquad$ {¬a, b}

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

{b}

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

{}

**Remark:** |=A iff there is **no** deduction of {} from $\mathbf{CL}_{\{\neg A\}}$

# Back To Arguments

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

"**We can**" deduce B from $A_1, \ldots\ldots., A_n$ means **validity** of argument

$$\frac{A_1, \ldots\ldots., A}{B}$$

**iff** by definition

$$|= (A_1 \wedge \ldots\ldots. \wedge A_n \Rightarrow B)$$

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

# Arguments

$\models (A_1 \wedge \ldots\ldots. \wedge A_n \Rightarrow B)$ **iff**

$=\!| \neg (A_1 \wedge \ldots\ldots. \wedge A_n \Rightarrow B)$ **iff**

$=\!| (A_1 \wedge \ldots\ldots. \wedge A_n \wedge \neg B)$

- **Step 1:** we transform $(A_1 \wedge .. \wedge A_n \wedge \neg B)$ to clauses
- Take A1,……., An and **find** $\mathbf{CL}_{A1}$ **, ….,** $\mathbf{CL}_{An}$

  and also **find** $\mathbf{CL}_{\neg B}$

  and **form**

  $\mathbf{CL}_{A1}$ U **….** $\mathbf{CL}_{An}$ U $\mathbf{CL}_{\neg B}$ **= CL**

**Step 2:** examine whether $\mathbf{CL} \vdash_R \{\}$

# Remember

- Argument $\frac{A1,\ldots\ldots, An}{B}$ is **valid** **iff**

$$\mathbf{CL}_{A1} \cup \ldots \cup \mathbf{CL}_{An} \cup \mathbf{CL}_{\neg B} \vdash_{\mathbf{R}} \{\}$$

$$\downarrow$$

Argument is **not valid**

**iff** **never** $\mathbf{CL}_{A1} \cup \ldots \cup \mathbf{CL}_{An} \cup \mathbf{CL}_{\neg B} \vdash_{\mathbf{R}} \{\}$

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

# Example

**Check** if you can deduce

$B = (\neg(a \cup \neg b) \Rightarrow (\neg a \wedge b))$

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

**Procedure:**

1. **Find** $\mathbf{CL}_{\{A1\}}$, $\mathbf{CL}_{\{A2\}}$ and $CL_{\{\neg B\}}$
2. **Form** $\mathbf{CL} = \mathbf{CL}_{\{A1\}} \cup \mathbf{CL}_{\{A2\}} \cup \mathbf{CL}_{\{\neg B\}}$
3. **Check** if $\mathbf{CL} \vdash_R \{\}$ or if never $\mathbf{CL} \vdash_R \{\}$

**Yes**, we can

**No**, we can't

# Example Solution

A1 = ((a => ¬b) => a)

¬(a=>¬b), a

a, a

¬¬b, a

b, a

**We get:** **CL**$_{A1}$ = {{a}, { b, a}}

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

¬a , (b=>a)

¬a, ¬b, a

**We get:** **CL**$_{A2}$ = {¬a, ¬b, a}

# Example Solution

- $\neg B = \neg(\neg(a \; U \neg b) \Rightarrow (\neg a \wedge b))$

$\neg(a \; U \neg b)$ $\qquad\qquad$ $\neg(\neg a \wedge b)$

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

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

**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}, {¬a, ¬b, c}, {¬a}, {b}, {a, ¬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.**

**CL** $\vdash$ **{}** **iff** **CL'** $\vdash$ **{}**

where **CL'** is obtained from **CL** by pure literal clauses **deletion**

# Example

- **CL** = {{¬a, ¬b, c}, {¬p, d}, {¬b, d}, {a}, {b}, {¬c}}

  d, ¬p are pure,

**CL’** = {{¬a, ¬b, c}, {a}, {b}, {¬c}}

{¬b, c}

{c}

{}

## S2. Tautology Deletion Strategy

- **Tautology –** a clause containing a **pair** of complementary literals ( a and ¬a)
- **S2: Tautology Deletion:**

  **CL'** = Remove all Tautologies from **CL**
- Example:
- **CL** = {{ a, b, ¬a}, {b, ¬b, c}, {a}}

  **CL'** = {{a}}
- Tautology Deletion Strategy S2 is **COMPLETE**.

**CL** is satisfiable **≡** **CL'** **is** satisfiable

**CL** unsatisfiable **≡** **CL'** unsatisfiable

# Exercise

- **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}}

{¬a} {¬b}

{b}

{ } **Efficient but not Complete!**

# Unit Resolution not complete Example

- **CL** = {{a, b}, {¬a, b}, {a, ¬b}, {¬a, ¬b}}

{b}

{a} {¬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** = $\{\{\neg a, c\}, \{\neg c\}, \{a, b\}, \{\neg b, c\}, \{\neg c\}\}$

$\{\neg a\}$

$\{b\}$

$\{c\}$

$\{\ \}$

**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**

# Input Resolution Deduction

Example: **CL** = {{a, b}, {¬a, c}, {¬b, c}, {¬c}}

{b, c}

{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}}

{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)