LECTURE 10
Chapter 10
CLASSICAL AUTOMATED PROOF SYSTEMS

PART 1: RS SYSTEM
PART 2: RS1, RS2, RS3 SYSTEMS
PART 3: GENTZEN SYSTEMS
Hilbert style systems are easy to define and admit a relatively simple proofs of the Completeness Theorem but they are difficult to use.

Automated systems are less intuitive then the Hilbert-style systems, but they will allow us to define effective automatic procedures for proof search, what is impossible in a case of the Hilbert-style systems.

The first idea of this type was presented by G. Gentzen in 1934.

We present in this chapter our version of original Gentzen system for propositional classical logic.

We present the original Gentzen systems for Intuitionistic and Classical Propositional Logics in Chapter 12.
PART 1: RS System

The automated proof system we presented here is due to Helena Rasiowa and Roman Sikorski. We present here the propositional version of the original system and call it RS system for Rasiowa - Sikorski.

The propositional RS system extends naturally to predicate logic QRS system which is presented in Chapter 14. Both systems RS and QRS admit a constructive proof of Completeness Theorem. First such constructive proofs were given, together with the formalization of the systems by H. Rasiowa and Sikorski in 1961.
AUTOMATED PROOF SYSTEMS

PART 2: RS1, RS2, RS3 System

We define, as an exercise 3 versions of the RS System, discuss their differences and show how the proof of Completeness Theorem for RS extends to similar proofs for all 3 systems.
AUTOMATED PROOF SYSTEMS

PART 3: GENTZEN Systems - Lecture 13

We present our modern versions of Gentzen Sequent systems for propositional classical logic.

Both systems extend easily to predicate logic and admit a constructive proof of Completeness Theorem via Rasiowa-Sikorski method.

The original Gentzen system LK for classical propositional logic is presented in Chapter 12 together with the original Gentzen system LI for the Intuitionistic propositional logic.
PART1: RS Proof System for Classical Propositional Logic
RS Proof System

Language of RS is

\[ \mathcal{L} = \mathcal{L}\{\neg, \Rightarrow, \cup, \cap}\]  

The rules of inference of our system RS operate on finite sequences of formulas and we adopt

\[ \mathcal{E} = \mathcal{F}^* \]

as the set of expressions of RS

Notation

Elements of \( \mathcal{E} \) are finite sequences of formulas and we denote them by

\[ \Gamma, \Delta, \Sigma \ldots \]

with indices if necessary.
RS Proof System

The intuitive meaning of a sequence $\Gamma \in F^*$ is that the truth assignment $v$ makes it true if and only if it makes the formula of the form of the disjunction of all formulas of $\Gamma$ true. For any sequence $\Gamma \in F^*$, we denote

$$\Gamma = A_1, A_2, ..., A_n$$

we denote

$$\delta_{\Gamma} = A_1 \cup A_2 \cup ... \cup A_n$$

We define as the next step a formal semantics for RS
Formal Semantics for RS

Let $v : VAR \rightarrow \{T, F\}$ be a truth assignment and $v^*$ its classical semantics extension to the set of formulas $\mathcal{F}$. We formally extend $v$ to the set $\mathcal{F}^*$ of all finite sequences of $\mathcal{F}$ as follows:

$$v^*(\Gamma) = v^*(\delta_\Gamma) = v^*(A_1) \cup v^*(A_2) \cup \ldots \cup v^*(A_n)$$

The sequence $\Gamma$ is said to be **satisfiable** if there is a truth assignment $v : VAR \rightarrow \{T, F\}$ such that $v^*(\Gamma) = T$. We write it as $v \models \Gamma$ and call $v$ a **model** for $\Gamma$. 
Formal Semantics for RS

The sequence $\Gamma$ is said to be **falsifiable** if there is a truth assignment $v$, such that $v^*(\Gamma) = F$

Such a truth assignment $v$ is called a **counter-model** for $\Gamma$

The sequence $\Gamma$ is said to be a **tautology** iff $v^*(\Gamma) = T$ for all truth assignments $v : \text{VAR} \rightarrow \{T, F\}$

We write as always,

$$\models \Gamma$$

to denote that $\Gamma$ is a **tautology**
Example

Let $\Gamma$ be a sequence

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

The truth assignment $v$ such that

$$v(a) = F \text{ and } v(b) = T$$

falsifies $\Gamma$, i.e. is a counter-model for $\Gamma$ as shows the following computation

$$v^*(\Gamma) = v^*(\delta_{\Gamma}) = v^*(a) \cup v^*(b \land a) \cup v^*(\neg b) \cup v^*(b \Rightarrow a) = F \cup (F \land T) \cup F \cup (T \Rightarrow F) = F \cup F \cup F \cup F = F.$$
Rules of inference

Rules of inference of RS are of the form:

\[ \Gamma_1 \quad \quad \text{or} \quad \quad \Gamma_1 ; \Gamma_2 \]
\[ \quad \Gamma \]

where \( \Gamma_1, \Gamma_2 \) are called \textit{premisses} and \( \Gamma \) is called the \textit{conclusion} of the rule.

Each rule of inference \textit{introduces} a new \textit{logical connective} or a \textit{negation} of a logical connective.

We \textit{name} the rule that introduces the logical connective \( \circ \) in the conclusion sequent \( \Gamma \) by \( (\circ) \).

The notation \( (\neg \circ) \) means that the \textit{negation} of the logical connective \( \circ \) is introduced in the conclusion sequence \( \Gamma \).
Rules of inference of RS

Proof System \( RS \) contains seven inference rules:

\[
(\cup), (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg\Rightarrow), (\neg\neg)
\]

Before we define the rules of inference of \( RS \) we need to introduce some definitions.

**Definition**

Any propositional variable, or a negation of propositional variable is called a **literal**

The set

\[
LT = VAR \cup \{\neg a : a \in VAR\}
\]

is called a set of all propositional **literals**

The variables are called **positive literals**

Negations of variables are called **negative literals**.
We denote by
\[ \Gamma', \Delta', \Sigma' \ldots \]
finite sequences (empty included) formed out of literals i.e
\[ \Gamma', \Delta', \Sigma' \in LT^* \]

We will denote by
\[ \Gamma, \Delta, \Sigma \ldots \]
the elements of \( F^* \)
Logical Axioms of RS

We adopt as an logical axiom of RS any sequence of literals which contains a propositional variable and its negation, i.e. any sequence

$$\Gamma_1', a, \Gamma_2', \neg a, \Gamma_3'$$

$$\Gamma_1', \neg a, \Gamma_2', a, \Gamma_3'$$

where \( a \in VAR \) is any propositional variable.

We denote by LA the set of all logical axioms of RS.
Inference Rules of RS

**Disjunction rules**

\[
(\cup) \quad \frac{\Gamma', A, B, \Delta}{\Gamma', (A \cup B), \Delta'} \quad (\neg\cup) \quad \frac{\Gamma', \neg A, \Delta; \Gamma', \neg B, \Delta}{\Gamma', \neg(A \cup B), \Delta}
\]

**Conjunction rules**

\[
(\cap) \quad \frac{\Gamma', A, \Delta; \Gamma', B, \Delta}{\Gamma', (A \cap B), \Delta} \quad (\neg\cap) \quad \frac{\Gamma', \neg A, \neg B, \Delta}{\Gamma', \neg(A \cap B), \Delta}
\]
Inference Rules of RS

Implication rules

\[
\begin{align*}
(\Rightarrow) & \quad \frac{\Gamma', \neg A, B, \Delta}{\Gamma', (A \Rightarrow B), \Delta'} \\
(\neg \Rightarrow) & \quad \frac{\Gamma', A, \Delta}{\Gamma', \neg (A \Rightarrow B), \Delta}
\end{align*}
\]

Negation rule

\[
(\neg \neg) \quad \frac{\Gamma', A, \Delta}{\Gamma', \neg \neg A, \Delta}
\]

where \( \Gamma' \in LT^*, \ \Delta \in F^*, \ A, B \in F \)
Proof System RS

Formally we define the system $\textbf{RS}$ as follows

$$\textbf{RS} = (\mathcal{L}_{\neg, \Rightarrow, \cup, \cap}, \mathcal{E}, \text{LA}, \mathcal{R})$$

where the set of inference rules is

$$\mathcal{R} = \{(\cup), (\neg \cup), (\cap), (\neg \cap), (\Rightarrow), (\neg \Rightarrow), (\neg \neg)\}$$

and $\text{LA}$ is the set of all logical axioms, as defined before.
Proof Trees

Definition

By a **proof tree** in RS of $\Gamma$ we understand a tree $T_\Gamma$

built out of sequences satisfying the following conditions:

1. The topmost sequence, i.e. the root of $T_\Gamma$ is the sequence $\Gamma$
2. all leafs are axioms
2. the nodes are sequences such that each sequence on the tree follows from the ones immediately preceding it by one of the inference rules
Proof Trees

We picture, and write our proof trees with the root on the top, and the leafs on the very bottom,

Additionally we write our proof trees indicating the name of the inference rule used at each step of the proof.

Example
Assume that a proof of a sequence $\Gamma$ from some three axioms was obtained by the subsequent use of the rules $(\cap), (\cup), (\cup), (\cap), (\cup)$, and $(\neg\neg), (\Rightarrow)$.

We represent it as the following tree.
Proof Trees

The tree $T_\Gamma$

\[
\Gamma
\]

\[
\vdash (\Rightarrow)
\]

\textit{conclusion of (\neg\neg)}

\[
\vdash (\neg\neg)
\]

\textit{conclusion of (\cup)}

\[
\vdash (\cup)
\]

\textit{conclusion of (\cap)}

\[
\bigwedge (\cap)
\]

\textit{conclusion of (\cap)}

\[
\vdash (\cup)
\]

\textit{conclusion of (\cup)}

\[
\vdash (\cup)
\]

\textit{axiom}

\[
\bigwedge (\cap)
\]

\textit{axiom}
Proof Trees

The **Proof Trees** represent a certain visualization for the proofs.

Any formal proof in any proof system can be represented in a tree form and vice-versa.

Any proof tree can be re-written in a linear form as a previously defined formal proof.

**Example**

The proof tree in RS of the de Morgan Law

\[ A = (\neg (a \cap b) \Rightarrow (\neg a \cup \neg b)) \]

is the as follows
Proof Trees

The tree $T_A$

$\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)$

$| (\Rightarrow)$

$\neg \neg (a \cap b), (\neg a \cup \neg b)$

$| (\neg \neg)$

$(a \cap b), (\neg a \cup \neg b)$

$\bigwedge (\cap)$

$a, (\neg a \cup \neg b)$  $b, (\neg a \cup \neg b)$

$| (\cup)$  $| (\cup)$

$a, \neg a, \neg b$  $b, \neg a, \neg b$
To obtain a formal proof (written in a vertical form) of \( A \) it we just write down the tree as a sequence, starting from the leaves and going up (from left to right) to the root.

\[
\begin{align*}
  & a, \neg a, \neg b \\
  & b, \neg a, \neg b \\
  & a, (\neg a \lor \neg b) \\
  & b, (\neg a \lor \neg b) \\
  & (a \land b), (\neg a \lor \neg b) \\
  & \neg \neg (a \land b), (\neg a \lor \neg b) \\
  & (\neg (a \land b) \Rightarrow (\neg a \lor \neg b))
\end{align*}
\]
Example

A search for the proof in RS of other de Morgan Law

\[ A = (\neg(a \cup b) \Rightarrow (\neg a \cap \neg b)) \]

consists of building a certain tree and proceeds as follows.
Example

The tree $T_A$

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

$| (\Rightarrow)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

$| (\neg)$$

$$(a \cup b), (\neg a \cap \neg b)$$

$| (\cup)$$

$$a, b, (\neg a \cap \neg b)$$

$\wedge (\cap)$

$a, b, \neg a$

$a, b, \neg b$
Example

We construct its formal proof, as before, written in a vertical manner.
Here it is

\[
\begin{align*}
a, b, \neg b \\
a, b, \neg a \\
a, b, (\neg a \cap \neg b) \\
(a \cup b), (\neg a \cap \neg b) \\
\neg \neg (a \cup b), (\neg a \cap \neg b) \\
(\neg (a \cup b) \Rightarrow (\neg a \cap \neg b))
\end{align*}
\]
Decomposition Trees

Our GOAL in inventing proof systems like RS is to facilitate automatic proof search.

The method of such proof search is to generate what is called the decomposition trees.

The decomposition tree for

\[ A = (((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) \]

is built as follows.
Decomposition Trees

The tree $T_A$

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$$

$$\cap (\cup)$$

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

$$\wedge (\cap)$$

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

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

$$\cap (\Rightarrow)$$

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

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

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

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

$$\neg a \land b, \neg a, c$$
Decomposition Trees

**Observe** that the decomposition tree $T_A$ contains an non-axiom leaf $\neg a, b, \neg a, c$

hence **it is not a proof** of $A$ in $RS$

Moreover, we are going to prove that a the decomposition trees in $RS$ are always **unique**

From the **uniqueness** of $T_A$ we have that if $T_A$ has a non-axiom leaf then the **proof** of $A$ in $RS$ **does not exist**

This fact becomes crucial in our proof of **Completeness** Theorem
Counter Models

The other crucial idea used in the proof of Completeness Theorem is that of a Counter Model defined by a decomposition tree.

Example Given a formula \( A \)

\[
((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)
\]

and its decomposition tree \( T_A \) presented on the next slide.
Counter Models

The tree $T_A$

\begin{align*}
(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) \\
\setminus (\cup) \\
((a \Rightarrow b) \cap \neg c), (a \Rightarrow c) \\
\wedge (\cap)
\end{align*}

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

\begin{align*}
\setminus (\Rightarrow) \\
\neg c, (a \Rightarrow c)
\end{align*}

\begin{align*}
\neg a, b, (a \Rightarrow c) \\
\setminus (\Rightarrow) \\
\neg c, \neg a, c
\end{align*}

\begin{align*}
\neg a, b, \neg a, c
\end{align*}
Counter Models

Consider a non-axiom leaf of $T_A$:

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

Let now $v$ be any variable assignment $v : \text{VAR} \rightarrow \{T, F\}$ such that it makes this non-axiom leaf false. We put constructing a $v$ restricted to $\text{VAR}_A$

$v(a) = T, v(b) = F, v(c) = F$

Obviously, we have that

$v^*(\neg a, b, \neg a, c) = v^*(\neg a) \cup v^*(b) \cup v^*(\neg a) \cup v^*(c) = F$
Moreover, we are going to prove that all the rules of inference of RS are strongly sound, i.e.

\[ C \equiv P, \quad C \equiv P_1 \cap P_2 \]

Strong soundness of the rules means that if at least one of premises of a rule is false, so is its conclusion.

We use the strong soundness of the rules to prove, by induction on the degree of sequences on a branch of \( T_A \) that starts with the formula \( A \) and ends with a non-axiom leaf, that any \( v \) that make this non-axiom leaf false also falsifies all sequences on the branch and hence falsifies the formula \( A \). This means that \( v \not\models A \), i.e. \( v \) is a counter-model for \( A \).
Consider a branch of $T_A$ with the non-axiom leaf $\neg a, b, \neg a, c$

In particular, the formula $A$ is on this branch, hence we get that
\[ v^*(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) = F \]

and $v$ is a counter-model for $A$

**Definition**

Any truth assignment that falsifies a non-axiom leaf is called a counter-model for $A$ generated by the decomposition tree $T_A$
Completeness Theorem

The construction of the counter-models generated by the decomposition trees is another crucial point to the proof of the Completeness Theorem for RS.

We prove first the following Completeness Theorems for formulas $A \in \mathcal{F}$

**Completeness Theorem 1** For any formula $A \in \mathcal{F}$

$$\vdash_{RS} A \quad \text{if and only if} \quad \models A$$

and then we generalize it to the following

**Completeness Theorem 2** For any $\Gamma \in \mathcal{F}^*$,

$$\vdash_{RS} \Gamma \quad \text{if and only if} \quad \models \Gamma$$
Strong Soundness of RS
Strong Soundness

Definition
Given a proof system

\[ S = (\mathcal{L}, \mathcal{E}, AX, \mathcal{R}) \]

Definition
A rule \( r \in \mathcal{R} \) such that the conjunction of all its premisses is logically equivalent to its conclusion is called strongly sound.

Definition
A proof system \( S \) is called strongly sound iff \( S \) is sound and all its rules \( r \in \mathcal{R} \) are strongly sound.
Strong Soundness of RS

Fact
The proof system RS is strongly sound

Proof
We prove as an example the strong soundness of two of inference rules: \((\cup)\) and \((\neg\cup)\)

Proof for all other rules follows the same patterns and is left as an exercise

By definition of strong soundness we have to show that
If \(P_1, P_2\) are premisses of a given rule and \(C\) is its conclusion, then for all \(v\),

\[ v^*(P_1) = v^*(C) \]

in case of one premiss rule and

\[ v^*(P_1) \cap v^*(P_2) = v^*(C) \]

in case of the two premisses rule.
Strong Soundness of RS

Consider the rule \((\cup)\)

\[
(\cup) \quad \frac{\Gamma', A, B, \Delta}{\Gamma', (A \cup B), \Delta}
\]

We evaluate:

\[
v^*(\Gamma', A, B, \Delta) = v^*(\delta_{\{\Gamma', A, B, \Delta\}}) = v^*(\Gamma') \cup v^*(A) \cup v^*(B) \cup v^*(\Delta)
\]

\[
= v^*(\Gamma') \cup v^*(A \cup B) \cup v^*(\Delta) = v^*(\delta_{\{\Gamma', (A \cup B), \Delta\}})
\]

\[
= v^*(\Gamma', (A \cup B), \Delta)
\]
Strong Soundness of RS

Consider the rule \((\neg \cup)\)

\[
\begin{array}{c}
(\neg \cup) \quad \frac{\Gamma', \neg A, \Delta}{\Gamma', \neg(B \cup A), \Delta}
\end{array}
\]

We evaluate:

\[
v^*(P_1) \cap v^*(P_2) = v^*(\Gamma', \neg A, \Delta) \cap v^*(\Gamma', \neg B, \Delta)
\]

\[
= (v^*(\Gamma') \cup v^*(\neg A) \cup v^*(\Delta)) \cap (v^*(\Gamma') \cup v^*(\neg B) \cup v^*(\Delta))
\]

\[
= (v^*(\Gamma', \Delta) \cup v^*(\neg A)) \cap (v^*(\Gamma', \Delta) \cup v^*(-B))
\]

\[
= \text{distrib} \quad (v^*(\Gamma', \Delta) \cup (v^*(\neg A) \cap v^*(\neg B))
\]

\[
= v^*(\Gamma') \cup v^*(\Delta) \cup v^*(\neg A \cap \neg B) = \text{deMorgan} \quad v^*(\delta_{\{\Gamma', \neg(A \cup B), \Delta\}}
\]

\[
= v^*(\Gamma', \neg(A \cup B), \Delta) = v^*(C)
\]
Soundness Theorem

Observe that the strong soundness notion implies soundness (not only by name!) and obviously all LA of RS are tautologies hence we have also proved the following

Soundness Theorem for RS

For any \( \Gamma \in \mathcal{F}^* \),

\[
\text{If } \vdash_{RS} \Gamma, \text{ then } \models \Gamma
\]

In particular, for any \( A \in \mathcal{F} \),

\[
\text{If } \vdash_{RS} A, \text{ then } \models A
\]
Completeness Theorem

Our goal now is to prove the **Completeness Part** of the **Completeness Theorem**, i.e. to prove that the following holds:

For any \( A \in \mathcal{F} \),

\[
\text{If } \models A, \text{ then } \vdash_{RS} A
\]

We prove instead the **opposite implication**

**RS Completeness Part**

\[
\text{If } \not\vdash_{RS} A \text{ then } \not\models A
\]

Here are main steps and facts needed for proof:

**Step 1** Define, for each \( A \in \mathcal{F} \) its decomposition tree \( T_A \)
Completeness Theorem

Step 2  Prove the following Lemmas

Lemma 1
For any $A \in \mathcal{F}$, the decomposition tree $T_A$ is unique

Lemma 2
For any $A \in \mathcal{F}$, $T_A$ has the following property:
$\kappa_{RS} A$ if and only if there is a leaf of $T_A$ which is not an axiom
Completeness Theorem

Lemma 3
For any $A \in \mathcal{F}$, such that $T_A$ has a non-axiom leaf, and for any truth assignment $v$, such that

$$v^*(\text{non-axiom leaf}) = F$$

the $v$ also falsifies $A$, i.e.

$$v^*(A) = F$$
Proof of Completeness Theorem

Proof of Completeness Theorem
Assume that $A$ is any formula such that

$\forall_{RS} A$

By Lemma 2 the decomposition tree $T_A$ contains a non-axiom leaf

The non-axiom leaf $L_A$ defines a truth assignment $v$ which falsifies $A$, as follows:

$$v(a) = \begin{cases} F & \text{if } a \text{ appears in } L_A \\ T & \text{if } \neg a \text{ appears in } L_A \\ \text{any value} & \text{if } a \text{ does not appear in } L_A \end{cases}$$

This proves Lemma 3 that

$\not\models A$
RS: DECOMPOSITION TREES
Decomposition Trees

The process of searching for a proof of a formula $A \in F$ in RS consists of building a certain tree $T_A$, called a decomposition tree.

Building a decomposition tree, i.e. a proof search tree consists in the first step of transforming the RS rules into corresponding decomposition rules.
RS Decomposition Rules

Here are all of RS decomposition rules

Disjunction decomposition rules

\[
(\cup) \quad \frac{\Gamma', (A \cup B), \Delta}{\Gamma', A, B, \Delta}, \quad \text{and} \quad (\neg \cup) \quad \frac{\Gamma', \neg (A \cup B), \Delta}{\Gamma', \neg A, \Delta; \Gamma', \neg B, \Delta}
\]

Conjunction decomposition rules

\[
(\cap) \quad \frac{\Gamma', (A \cap B), \Delta}{\Gamma', A, \Delta; \Gamma', B, \Delta'}, \quad \text{and} \quad (\neg \cap) \quad \frac{\Gamma', \neg (A \cap B), \Delta}{\Gamma', \neg A, \neg B, \Delta}
\]
Decomposition Rules

Implication decomposition rules

\[
\frac{\Gamma', (A \Rightarrow B), \Delta}{\Gamma', \neg A, B, \Delta}, \quad \frac{\Gamma', \neg(A \Rightarrow B), \Delta}{\Gamma', A, \Delta; \Gamma', \neg B, \Delta}
\]

Negation decomposition rule

\[
\frac{\Gamma', \neg\neg A, \Delta}{\Gamma', A, \Delta}
\]

where \( \Gamma' \in \mathcal{F}'^* \), \( \Delta \in \mathcal{F}^* \), \( A, B \in \mathcal{F} \)
Tree Decomposition Rules

We write the decomposition rules in a visual tree form as follows

Tree Decomposition Rules

∪ rule

\[ \Gamma', (A \cup B), \Delta \]

\[ l(\cup) \]

\[ \Gamma', A, B, \Delta \]
Tree Decomposition Rules

(¬∪) rule

\[ \Gamma', \neg(A \cup B), \Delta \]

\[ \bigwedge (¬∪) \]

(∩) rule

\[ \Gamma', \neg A, \Delta \]

\[ \Gamma', \neg B, \Delta \]

\[ \bigwedge (\cap) \]

\[ \Gamma', (A \cap B), \Delta \]

\[ \bigwedge (\cap) \]

\[ \Gamma', A, \Delta \]

\[ \Gamma', B, \Delta \]
Tree Decomposition Rules

(¬∪) rule

Γ′, ¬(A ∩ B), Δ

| (¬∩)

Γ′, ¬A, ¬B, Δ

(⇒) rule

Γ′, (A ⇒ B), Δ

| (∪)

Γ′, ¬A, B, Δ
Tree Decomposition Rules

(\neg \Rightarrow) rule

\Gamma', \neg (A \Rightarrow B), \Delta

\bigwedge (\neg \Rightarrow)

\Gamma', A, \Delta \quad \Gamma', \neg B, \Delta

(\neg \neg) rule

\Gamma', \neg

\neg A, \Delta

\neg (\neg \neg)

\Gamma', A, \Delta
Definitions and Observations

Observe that we use the same names for the inference and decomposition rules, as once the we have built the decomposition tree with all leaves being axioms, it constitutes a proof of $A$ in $RS$ with branches labeled by the proper inference rules.

Now we still need to introduce few standard and useful definitions and observations.

Definition: Indecomposable Sequence

A sequence $\Gamma'$ built only out of literals, i.e. $\Gamma \in \mathcal{F}'^*$ is called an indecomposable sequence.
Definitions and Observations

Definition: Decomposable Formula
A formula that is not a literal, i.e., $A \in F - LT$ is called a decomposable formula.

Definition: Decomposable Sequence
A sequence $\Gamma$ that contains a decomposable formula is called a decomposable sequence.
Definitions and Observations

Observation 1
Decomposition rules are functions with disjoint domains, i.e.

For any decomposable sequence, i.e. for any $\Gamma \notin LT^*$ there is exactly one decomposition rule that can be applied to it

This rule is determined by the first decomposable formula in $\Gamma$ and by the main connective of that formula
Definitions and Observations

Observation 2
If the main connective of the first decomposable formula is \( \cup, \cap, \Rightarrow \), then the decomposition rule determined by it is \((\cup), (\cap), (\Rightarrow)\), respectively.

Observation 3
If the main connective of the first decomposable formula \( A \) is negation \( \neg \), then the decomposition rule is determined by the second connective of the formula \( A \). The corresponding decomposition rules are \((\neg \cup), (\neg \cap), (\neg \neg), (\neg \Rightarrow)\).
Lemma

Because of the importance of the Observation 1 we re-write it in a form of the following

**Unique Decomposition Lemma**

For any sequence $\Gamma \in \mathcal{F}^*$,

$\Gamma \in \mathcal{LT}^*$ or $\Gamma$ is in the domain of **exactly one** of RS

Decomposition Rules
Definition: Decomposition Tree $T_A$

For each $A \in \mathcal{F}$, a decomposition tree $T_A$ is a tree built as follows:

**Step 1.**

The formula $A$ is the root of $T_A$.

For any other node $\Gamma$ of the tree we follow the steps below:

**Step 2.**

If $\Gamma$ is indecomposable then $\Gamma$ becomes a leaf of the tree.
Decomposition Tree Definition

Step 3.
If $\Gamma$ is decomposable, then we traverse $\Gamma$ from left to right and identify the first decomposable formula $B$.

By the Unique Decomposition Lemma and Observations 2,3 there is exactly one decomposition rule determined by the main connective of $B$.

We put its premiss as a node below, or its left and right premisses as the left and right nodes below, respectively.

Step 4.
We repeat steps 2 and 3 until we obtain only leaves.
Decomposition Theorem

We now our Lemmas 1, 2, 3 needed for the proof of the Completeness Theorem into one

Decomposition Tree Theorem

For any sequence $\Gamma \in \mathcal{F}^*$ the following conditions hold

1. $T_\Gamma$ is finite and unique
2. $T_\Gamma$ is a proof of $\Gamma$ in $\text{RS}$ if and only if all its leafs are axioms
3. $\not\Gamma_{\text{RS}}$ if and only if $T_\Gamma$ has a non-axiom leaf
Theorem

Proof
The tree $T_{\Gamma}$ is unique by the Unique Decomposition Lemma.

It is finite because there is a finite number of logical connectives in $\Gamma$ and all decomposition rules diminish the number of connectives.

If the tree $T_{\Gamma}$ has a non-axiom leaf it is not a proof by definition.

By 1. it also means that the proof does not exist.
Example

Let's construct, as an example a decomposition tree $T_A$ of the following formula $A$

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

The formula $A$ forms a one element decomposable sequence

The first decomposition rule used is determined by its main connective

We put a box around it, to make it more visible

$$((a \cup b) \Rightarrow \neg a) \boxed{\cup}(\neg a \Rightarrow \neg c))$$
Example

The first and only decomposition rule to be applied is \((\cup)\)

The first segment of the decomposition tree \(T_A\) is

\[
((a \cup b) \Rightarrow \neg a) | (\neg a \Rightarrow \neg c))
\]

| \((\cup)\) |
| \((\cup)\) |

\[
((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c)
\]
Example

Now we decompose the sequence

\[ ((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c) \]

It is a decomposable sequence with the first, decomposable formula

\[ ((a \cup b) \Rightarrow \neg a) \]

The next step of the construction of our decomposition tree is determined by its main connective \( \Rightarrow \) and we put the box around it

\[ ((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c) \]
Example

The decomposition tree becomes now

\(( (a \cup b) \Rightarrow \neg a ) \cup (\neg a \Rightarrow \neg c) \)

\[ \begin{align*}
| & (\cup) \\
| & ((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c) \\
| & (\Rightarrow) \\
\neg (a \cup b), \neg a, (\neg a \Rightarrow \neg c)
\end{align*} \]
Example

The next sequence to decompose is

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

with the first decomposable formula

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

Its main connective is \( \neg \), so to find the appropriate rule we have to examine next connective, which is \( \cup \)

The \textbf{decomposition rule} determine by this stage of decomposition is \( (\neg \cup) \)
Example

Next stage of the construction of the decomposition tree $T_A$ is

$$((a \cup b) \Rightarrow \neg a) \bigcup (\neg a \Rightarrow \neg c))$$

| (\bigcup)

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

| (\Rightarrow)

\square (a \cup b), \neg a, (\neg a \Rightarrow \neg c)

\bigwedge (\neg \bigcup)

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

Finally, the complete $T_A$ is

\[
((a \cup b) \Rightarrow \neg a) \cup (\neg a \Rightarrow \neg c))
\]

\[
\bigl( (a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c) \bigr)
\]

\[
\bigl( \neg(a \cup b), \neg a, (\neg a \Rightarrow \neg c) \bigr)
\]

\[
\neg a, \neg a, (\neg a \Rightarrow \neg c)
\]

\[
\neg b, \neg a, (\neg a \Rightarrow \neg c)
\]
Example

All leaves of $T_A$ are axioms

The tree $T_A$ is a proof of $A$ in $RS$, i.e.

$$\vdash_{RS} ((a \cup b) \Rightarrow \neg a) \cup (\neg a \Rightarrow \neg c)$$
Example

Example Given a formula $A$ and its decomposition tree $T_A$

$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$

$\mid (\cup)$

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

$\wedge (\cap)$

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

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

$\mid (\Rightarrow)$

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

$\neg c, \neg a, c$

$\mid (\Rightarrow)$

$\neg a, b, \neg a, c$
Counter Model

Consider a non-axiom leaf of $T_A$

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

We will define now define a counter-model generated by a decomposition tree $T_A$

$v$, by definition is any variable assignment

$v : \text{VAR} \rightarrow \{T, F\}$

that makes this non-axiom leaf false i.e. for example we put

$v(a) = T, v(b) = F, v(c) = F$

Obviously, we have that

$v^*(-a, b, \neg a, c) = \neg T \cup F \cup \neg T \cup F = F$
Counter Model

We have **proved** that **RS** is **strongly sound**.

The **strong soundness** of the rules means that if **one of premisses** of a rule is **false**, so is the **conclusion**.

Hence, the **strong soundness** of the rules **proves**, by induction on the degree of sequences $\Gamma \in T_A$, which $\nu$ that made a leaf **false** **falsifies** all sequences on the **branch** of $T_A$ that ends with the already **falsified** leaf.
Counter Model Theorem

We have hence proved the following

**Counter Model Theorem**

Let \( A \in \mathcal{F} \) be such that its decomposition tree \( T_A \) contains a **non-axiom** leaf \( L_A \).

Any truth assignment \( v \) that **falsifies** \( L_A \) is a **counter model** for \( A \).
Counter Model

In particular, the formula \( A \) belongs to the branch with falsified non-axiom leaf

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

By the Counter Model Theorem

\[
\nu^*(A) = \nu^*((((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) = F
\]

i.e. \( \nu \) is a counter-model for \( A \) and we proved that

\( \not \models A \)
Counter Model

F "climbs" the tree $T_A$

\[((((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) = F\]

\[\neg c, (a \Rightarrow c)\]

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

axiom

\[
\neg a, b, (a \Rightarrow c) = F
\]

\[\neg a, b, \neg a, c = F\]
Counter Model

**Observe** that the same *counter model construction* applies to any other *non-axiom leaf*, if exists.

The other *non-axiom leaf* gives the other *F* climbs the tree picture, and hence another *counter-model* for *A*.

By *Decomposition Tree Theorem* all possible restricted *counter-models* for *A* are those generated by all *non-axioms* leaves of the *T_A*.

In our case the formula *A* has only *one non-axiom leaf*, and hence only one restricted *counter model*. 
Completeness Theorem Revisited

RS Completeness Theorem
For any $A \in \mathcal{F}$,

If $\models A$, then $\vdash_{RS} A$

We prove instead the opposite implication

RS Completeness Theorem

If $\not\vdash_{RS} A$ then $\not\models A$
Proof of Completeness Theorem

Proof of Completeness Theorem

Assume that $A$ is any formula is such that $\not\models_{RS} A$

By the **Decomposition Tree Theorem** the $T_A$ contains a non-axiom leaf.

The non-axiom leaf $L_A$ **defines** a truth assignment $v$ which **falsifies** it as follows:

$$v(a) = \begin{cases} 
F & \text{if } a \text{ appears in } L_A \\
T & \text{if } \neg a \text{ appears in } L_A \\
\text{any value} & \text{if } a \text{ does not appear in } L_A 
\end{cases}$$

Hence by **Counter Model Theorem** we have that $v$ also **falsifies** $A$, i.e.

$\not\models A$
PART2:
RS1, RS2, RS3 Proof Systems
RS1 Proof System

Language of RS1 is the same as the language of RS, i.e.

\[ \mathcal{L} = \mathcal{L}\{\neg, \Rightarrow, \cup, \cap\} \]

The rules of inference of our system RS1 operate as rules of RS on finite sequences of formulas and we adopt

\[ \mathcal{E} = \mathcal{F}^* \]

as the set of expressions of RS1

Notation

Elements of \( \mathcal{E} \) are finite sequences of formulas and we denote them by

\[ \Gamma, \Delta, \Sigma \ldots \]

with indices if necessary.
Rules of inference of RS1

Proof System  RS1  contains seven inference rules, denoted by the same symbols as the rules of RS

(∪), (¬∪), (∩), (¬∩), (⇒), (¬⇒), (¬¬)

The inference rules of RS1 are quite similar to the rules of RS - look at them CAREFULLY! to see where lies the difference!

REMINDER: Definition

Any propositional variable, or a negation of propositional variable is called a literal

The set

\[ LT = VAR \cup \{\neg a : \ a \in VAR\} \]

is called a set of all propositional literals

The variables are called positive literals

Negations of variables are called negative literals.
Literals Notation

We denote, as before, by

$$\Gamma', \Delta', \Sigma' \ldots$$

finite sequences (empty included) formed out of literals i.e

$$\Gamma', \Delta', \Sigma' \in LT^*$$

We will denote by

$$\Gamma, \Delta, \Sigma \ldots$$

the elements of $$F^*$$
Logical Axioms of RS1

We adopt all logical axiom of RS as the axioms of RS1, i.e.

Logical Axioms LA of RS1 are as follows

$$\Gamma_1', \ a, \ \Gamma_2', \ \neg a, \ \Gamma_3'$$

$$\Gamma_1', \ \neg a, \ \Gamma_2', \ a, \ \Gamma_3'$$

where $a \in VAR$ is any propositional variable
Inference Rules of RS1

Disjunction rules

\[
\begin{align*}
(\cup) & \quad \frac{\Gamma, \ A, B, \Delta'}{\Gamma, \ (A \cup B), \Delta'} \\
(-\cup) & \quad \frac{\Gamma, \neg A, \Delta'; \Gamma, \neg B, \Delta'}{\Gamma, \neg(A \cup B), \Delta'}
\end{align*}
\]

Conjunction rules

\[
\begin{align*}
(\cap) & \quad \frac{\Gamma, A, \Delta'; \Gamma, B, \Delta'}{\Gamma, \ (A \cap B), \Delta'} \\
(-\cap) & \quad \frac{\Gamma, \neg A, \neg B, \Delta'}{\Gamma, \neg(A \cap B), \Delta'}
\end{align*}
\]
Inference Rules of RS1

Implication rules

\[
\begin{align*}
(\Rightarrow) & \quad \frac{\Gamma, \neg A, B, \Delta'}{\Gamma, (A \Rightarrow B), \Delta'} \\
(\neg \Rightarrow) & \quad \frac{\Gamma, A, \Delta'}{\Gamma, \neg (A \Rightarrow B), \Delta'}
\end{align*}
\]

Negation rule

\[
\begin{align*}
(\neg \neg) & \quad \frac{\Gamma, A, \Delta'}{\Gamma, \neg \neg A, \Delta'}
\end{align*}
\]

where \( \Gamma' \in LT^*, \Delta \in \mathcal{F}^*, A, B \in \mathcal{F} \)
Proof System RS1

Formally we define the system $\text{RS1}$ as follows

$$\text{RS1} = (\mathcal{L}_{\neg, \Rightarrow, \cup, \cap}, \mathcal{E}, \text{LA}, \mathcal{R})$$

where

$$\mathcal{R} = \{(\cup), (\neg \cup), (\cap), (\neg \cap), (\Rightarrow), (\neg \Rightarrow), (\neg\neg)\}$$

for the inference rules is defined above and $\text{LA}$ is the set of all logical axioms (the same as for $\text{RS}$)
System RS1

Exercise
1. Construct a proof in RS1 of a formula

\[ A = (\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)) \]

2. Prove that RS1 is strongly sound

3. Define in your own words, for any formula A, the decomposition tree \( T_A \) in RS1

4. Prove Completeness Theorem for RS1
System RS1

The decomposition tree $T_A$ in RS1 is a proof of $A$ in RS1 as all leaves are axioms

$$T_A
\begin{align*}
(\neg (a \cap b) &\Rightarrow (\neg a \cup \neg b)) \\
| (\Rightarrow) \\
(\neg \neg (a \cap b), (\neg a \cup \neg b) \\
| (\cup) \\
\neg \neg (a \cap b), \neg a, \neg b \\
| (\neg) \\
(a \cap b), \neg a, \neg b \\
\wedge (\cap)
\end{align*}$$

$a, \neg a, \neg b$ $b, \neg a, \neg b$
2. **Observe** that the system **RS1** is obtained from **RS** by changing the sequence $\Gamma'$ into $\Gamma$ and the sequence $\Delta$ into $\Delta'$ in all of the rules of inference of **RS**

These changes do not influence the essence of proof of strong soundness of the rules of **RS**

One has just to replace the sequence $\Gamma'$ by $\Gamma$ and $\Delta$ by $\Delta'$ in the proof of strong soundness of each rule of **RS** to obtain the corresponding proof of strong soundness of corresponding rule of **RS1**

We do it, for example for the rule $(\cup)$ of **RS1** as follows
Strong Soundness of RS1

Consider the rule \((\cup)\) of RS1

\[
(\cup) \quad \frac{\Gamma, A, B, \Delta'}{\Gamma, (A \cup B), \Delta'}
\]

We evaluate:

\[
v^*(\Gamma, A, B, \Delta') = v^*(\delta_{\{\Gamma, A, B, \Delta'\}}) = v^*(\Gamma) \cup v^*(A) \cup v^*(B) \cup v^*(\Delta')
\]

\[
= v^*(\Gamma) \cup v^*(A \cup B) \cup v^*(\Delta') = v^*(\delta_{\{\Gamma, (A \cup B), \Delta'\}})
\]

\[
= v^*(\Gamma, (A \cup B), \Delta')
\]
3. The definition of the decomposition tree $T_A$ is again, it its essence similar to the one for RS except for the changes which reflect the differences in the corresponding rules of inference.

We follow now the following steps

**Step 1**
Decompose using rule defined by the main connective of a decomposable formula $B$

**Step 2**
Traverse resulting sequence $\Gamma$ on the new node of the tree from RIGHT to LEFT and find first decomposable formula

**Step 3**
Repeat **Step 1** and **Step 2** until no more decomposable formulas

**End of Tree Construction**
Decomposition Trees in RS1

4.

Observe that directly from the definition of the decomposition tree $T_A$ we have that the following holds

Fact 1: The decomposition tree $T_A$ is a proof iff all leaves are axioms

Fact 2: The proof does not exist otherwise, i.e. $\not\models_{RS1} A$ iff there is a non-axiom leaf on $T_A$

Fact 2 holds because the tree because the tree $T_A$ is unique

Observe that we need Facts 1, 2 in order to prove Completeness Theorem by construction of a counter-model generated by a non-axiom leaf
Proof of Completeness Theorem for RS1

Proof of Completeness Theorem
Assume that $A$ is any formula is such that

$\not\models_{RS1} A$

By Fact 2 the decomposition tree $T_A$ contains a non-axiom leaf.

The non-axiom leaf $L_A$ defines a truth assignment $v$ which falsifies $A$, as follows:

$$v(a) = \begin{cases} 
  F & \text{if } a \text{ appears in } L_A \\
  T & \text{if } \neg a \text{ appears in } L_A \\
  \text{any value} & \text{if } a \text{ does not appear in } L_A
\end{cases}$$

This proves that

$\not\models A$
System RS2

Definition
System RS2 is a proof system obtained from RS by changing the sequences $\Gamma'$ into $\Gamma$ in all of the rules of inference of RS. The logical axioms LA remind the same.

Exercises
E1 Construct two decomposition trees in RS2 of the formula

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

E2 Show that RS2 is strongly sound
E3 Prove the Soundness Theorem for RS2
Exercises

E3  Define shortly, in your own words, for any formula $A$, its decomposition tree $T_A$ in $RS2$

Justify why your definition is correct

Show that in $RS2$ the decomposition tree for as given formula $A$ may not be unique

E4  Prove the Completeness Theorem for $RS2$
System RS2

Exercise

Write a procedure $\text{TREE}_A$ such that for any formula $A$ of RS2 it produces its UNIQUE decomposition tree and prove COMPLETENESS of this procedure
System RS3

Definition
System RS2 is a proof system obtained from RS by changing its LA to the following set of axioms.
The rules of inference remind the same

\[ \Gamma_1, A, \Gamma_2, \neg A, \Gamma_3 \]

\[ \Gamma_1, \neg A, \Gamma_2, A, \Gamma_3 \]

where \( A \in \mathcal{F} \) is any formula.
We denote by LA the set of all logical axioms of RS3.
System RS3

Prove the **Completeness Theorem** for RS3