

CSE541 MIDTERM 2 SOLUTIONS
(as submitted by Kuznetsova Polina)

Question 1

Given a set

$$S = \{((a \cap b) \Rightarrow b), (a \cup b), \neg a\}.$$

1. Show that S is **consistent**.
2. Show that a formula $A = (\neg a \cap b)$ is **not independent** of S .
3. Find an infinite number of formulas that are independent of S .
4. Give an example of **an infinite consistent** set S (propositional language).

Solution

$$S = \{((a \cap b) \Rightarrow b), (a \cup b), \neg a\}$$

1. We need to find a model for S , i.e. M , such that M satisfies all the formulas in S .

For example, $M=v$, where v is any $v : VAR \rightarrow \{t, F\}$ such that $v(a)=F$, $v(b)=T$.

$$v * (((a \cap b) \Rightarrow b)) = ((F \cap T) \Rightarrow T) = (F \Rightarrow T) = T$$

$$v * (a \cup b) = (F \cup T) = T$$

$$v * (\neg a) = \neg F = T$$

I.e. S has a model, thus, S is consistent

2. $A = (\neg a \cap b)$

To show that A is not independent of S, we need to show that either $S \cup \{A\}$ or $S \cup \{\neg A\}$ is inconsistent. $\neg A = a \cup \neg b$

Notice that for S to be consistent we need a to be F (since $\neg a \in S$).

Also we need b to be T, since a is F and $(a \cup b) \in S$.

But if $v(a)=F$, $v(b)=T$, then $\neg A = F \cup \neg T = F \cup F = F$.

Thus, $S \cup \{\neg A\}$ is inconsistent and so A is not independent of S.

3. Any formula that does not contain a,b is independent of S, since its model includes all possible truth assignments for variables a,b.

Thus, a sequence of formulas a_1, a_2, a_3, \dots , where a_i is a variable and $a_i \neq a$, $a_i \neq b$, is independent from S.

4. We can take $S = \{a, (a \cap a_1), (a \cap a_2), (a \cap a_3), \dots\}$, where a, a_i are variables.

A truth assignment that have $v(a)=T$, $v(a_i)=T$ is a model for S, thus S is consistent, since it has a model.

Question 2

Given a set S of formulas:

$$S = \{\forall x((R(x, y) \cap R(y, z)) \Rightarrow R(x, z)), \forall xR(x, x)\}.$$

Remember: $R(x, y)$ is a two argument predicate representing a binary relation.

1. Show that S is **consistent**.

2. Show that a formula $A = \forall x(R(x, y) \Rightarrow R(y, x))$ is **independent** of S.

Solution

$$S = \{\forall x((R(x, y) \cap R(y, z)) \Rightarrow R(x, z)), \forall xR(x, x)\}$$

1. We need to find a structure for S that is a model.

Define $M = \{\mathbb{N}, \leq, \emptyset\}$

Then, $S = \{\forall x((x \leq y) \cap (y \leq z)) \Rightarrow (x \leq z), \forall x(x \leq x)\}$

I.e. S consists of true statements, thus M is a model for S.

Thus, S is consistent.

2. $A = \forall x(R(x, y) \Rightarrow R(y, x))$

To show that A is independent of S we need to show that $S \cup \{A\}$ and $S \cup \{\neg A\}$ are both consistent.

$\neg A = \neg(\forall x(R(x, y) \Rightarrow R(y, x))) = \exists x \neg(R(x, y) \Rightarrow R(y, x)) =$

$\exists x(R(x, y) \cap \neg R(y, x))$

Consider $S_1 = S \cup \{A\}$

Take $M_1 = \{\mathbb{N}, =, \emptyset\}$

$S_1 = \{\forall x(((x = y) \cap (y = z)) \Rightarrow (x = z)), \forall x(x = x), \forall x((x = y) \Rightarrow (y = x))\}$

S_1 consists of truth statements, thus M_1 is a model for S_1 . Thus, S_1 is consistent. Consider $S_2 = S \cup \{\neg A\}$

Take $M_2 = \{\mathbb{N}, \leq, \emptyset\}$

$S_2 = \{\forall x(((x \leq y) \cap (y \leq z)) \Rightarrow (x \leq z)), \forall x(x \leq x), \exists x((x \leq y) \cap \neg(y \leq x))\}$

$S_2 = \{\forall x(((x \leq y) \cap (y \leq z)) \Rightarrow (x \leq z)), \forall x(x \leq x), \exists x((x \leq y) \cap (y > x))\}$

S_2 consists of truth statements, thus M_2 is a model for S_2 . Thus, S_2 is consistent. Thus, A is independent of S.

Question 3

1. Show that if $S = \emptyset$, then for any formula A of \mathcal{F} of propositional or predicate language,

$$S \models A \text{ implies that } \models A.$$

2. Show that there is $S \neq \emptyset$, such that for any A , such that $\models A$, $S \models A$.
3. Show that if $S \subseteq \mathcal{F}$ is **inconsistent** then $\{A : S \models A\} = \mathcal{F}$.

Solution

S semantically entails A ($S \models A$), if for any model $M \models S$, M also is a model for A ($M \models A$).

Thus, if we denote $MOD(S)$ and $MOD(A)$ as set of all models for S and set of all models of A respectively, i.e.

$$MOD(S) = \{M : M \models S\}, \quad MOD(A) = \{M : M \models A\}$$

then we can re-write definition of the semantic entailment as follows:

$$\text{If } S \models A \text{ then } MOD(S) \subseteq MOD(A).$$

1. $S = \emptyset, S \models A$

Since S is empty, Any model M satisfies it. I.e. $MOD(S)$ is a set of all possible truth assignments (or structures in case of predicate logic).

Consider a formula A, such that $S \models A$, i.e. $MOD(S) \subseteq MOD(A)$.

But if $MOD(S)$ is a set of all possible models, then its superset $MOD(A)$ is also a set of all possible models. Thus any model satisfies A.

I.e. $\models A$.

2. We need to find S such that for any A if $\models A$ then $S \models A$. I.e. we just need to find a set of formulas S such that $MOD(S) \subseteq MOD(A)$ for any tautology A. $MOD(A)$ consists of all possible truth assignment over set of variables VAR. So any S will have $MOD(S) \subseteq MOD(A)$, for example S consisting of one formula $S = \{a\}$, where a is a variable.

3. If $S \subseteq F$ is inconsistent, then $MOD(S) = \emptyset$. But any formula in F has either no models or at least one model. And empty set is a subset of any set. So for any formula $A \in F$ $MOD(S) \subseteq MOD(A)$.

Thus, $\forall A \in FS \models A$ and $\{A : S \models A\} = F$

Question 4

Consider a system **RS1** obtained from **RS** by changing the sequence Γ' into Γ in all of the rules of inference of **RS**.

1. Explain why the system **RS1** is sound. You can use the Soundness of the system **RS**.

2. Construct **TWO** decomposition trees of

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

3. If there is a tree constructed that is not a proof, construct the counter-model determined by that tree. Justify that it is a counter-model.

Solution

1. The only one difference between RS and RS1 is that for RS1 in each inference rule at the beginning there are sequence of formulas, not just literals.

So there are many ways to apply rules while constructing tree, but it does not affect soundness, since for all rules premisses and conclusions still logically equivalent.

Consider, for example, rule \cup . In RS1 it is:

$$\frac{\Gamma, A, B, \Delta}{\Gamma, (A \cup B), \Delta}$$

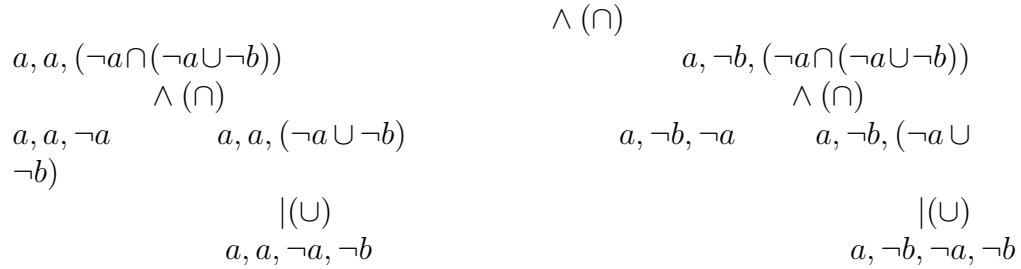
$$\begin{aligned} v * (\Gamma, A, B, \Delta) &= v * (\Gamma) \cup v * (A) \cup v * (B) \cup v * (\Delta) = v * (\Gamma) \cup \\ &v * (A \cup B) \cup v * (\Delta) = \\ &= v * (\Gamma, (A \cup B), \Delta) \end{aligned}$$

For all rules premisses and conclusion are logically equivalent, thus RS1 is sound.

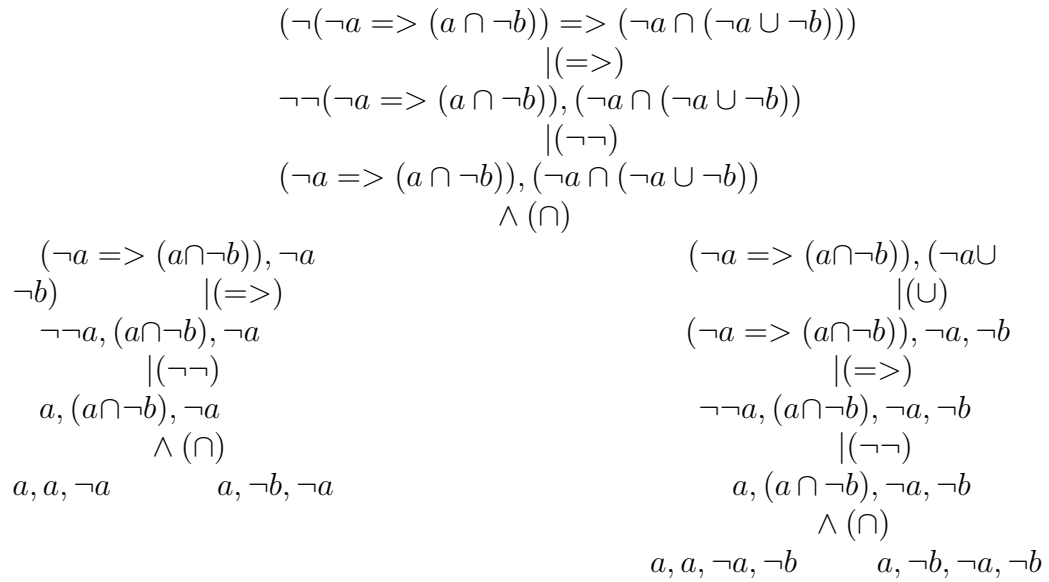
2. $(\neg(\neg a \Rightarrow (a \cap \neg b)) \Rightarrow (\neg a \cap (\neg a \cup \neg b)))$

Tree1

$$\begin{aligned} &(\neg(\neg a \Rightarrow (a \cap \neg b)) \Rightarrow (\neg a \cap (\neg a \cup \neg b))) \\ &\quad |(\Rightarrow) \\ &\neg\neg(\neg a \Rightarrow (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b)) \\ &\quad |(\neg\neg) \\ &(\neg a \Rightarrow (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b)) \\ &\quad |(\Rightarrow) \\ &\neg\neg a, (a \cap \neg b), (\neg a \cap (\neg a \cup \neg b)) \\ &\quad |(\neg\neg) \\ &a, (a \cap \neg b), (\neg a \cap (\neg a \cup \neg b)) \end{aligned}$$



Tree2



3. Both trees have axioms as leaves. So both trees are proofs.

Question 5

1. Define shortly, in your own words, for any A , the decomposition tree \mathbf{T}_A in **RS1** as defined in QUESTION 3. Justify why your definition is

correct.

Show that in **RS1** decomposition tree may not be unique.

2. Prove the **Completeness Theorem** for **RS1** (do not need to prove soundness).

Solution

1. For a formula A decomposition tree T_A can be defined as following. It has A as a root. For each node, if there is a rule of RS1 which conclusion has the same form as node formulas, thus node has children that are premises of the rule.

If the node consists only of atomic formulas (i.e. no rules can be apply), then it does not have any children.

The last statement define a termination condition for a tree.

This definition correctly defines decomposition tree for a formula as a proof for thus formula, since it uses the rules, which are sound (strongly sound).

Since for RS1 all rules of inference have Γ instead of Γ' as in RS, the choice of the rule for a node is not unique. For example consider a node $(a \Rightarrow b), (b \cup a)$. Γ is a set of formulas, not literals, so for this node we can choose either \Rightarrow or \cup rule. This lead to a non-unique tree.

2. We need to prove that for a formula A if $\not\vdash_{RS1} A$ then $\not\models A$
If $\not\vdash_{RS1} A$ then every tree of A has at least one non-axiom leaf.
Otherwise, there would exist tree with all leafs as axioms and it would be a proof for A.

Thus, we have a tree T_A with at least one non-axiom leaf.

We can construct a counter-model for A in the following way. We take non-axiom leaf L_A and for each variable assign a value:

$$v(a) = \begin{cases} F, & \text{if } a \text{ appears in } L_A \\ T, & \text{if } \neg a \text{ appears in } L_A \\ \text{any value,} & \text{if neither } a \text{ or } \neg a \text{ appears in } L_A \end{cases}$$

The value for a formula that corresponds to a leaf in this case will be F. Since, because of the strong soundness F "climbs" the tree, we found a counter-model for A.

Thus, $\not\models A$

Question 6

Write a procedure $TREE_A$ that for any formula A of **RS1** it produces its UNIQUE decomposition tree and prove **COMPLETENESS** of this procedure.

Solution

```
procedure  $TREE_A$ (Formula A, Tree T)
{
   $B = ChoseLeftMostFormula(A)$  // Choose the left most formula
  that is not a literal
   $c = MainConnective(B)$  // Find the main connective of B
   $R = FindRule(c)$  // Find the rule which conclusion that has this
  connective
   $P = Premises(R)$  // Get the premises for this rule
   $AddToTree(A, P)$  // add premises as children of A to the tree
  Forall p in P // go through all premises
     $TREE_A(p, T)$  // build subtrees for each premis
}
```

Procedure provides unique tree, since it always chooses the most left formula for a rule.

This procedure is equivalent to RS system, since with rules of RS the most left formula is always chosen.

RS system is complete, thus this procedure is complete.

Question 7

1. Let **GL** be the Gentzen style proof system defined in chapter 11.

1. Prove, by constructing a proper decomposition trees that

$$\not\vdash_{\mathbf{GL}}((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).$$

2 Use the above to prove, without use of the Completeness Theorem that

$$\not\vdash ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).$$

Solution

1. $A = ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$

We need to construct all possible trees to show that A does not have a proof in **GL**.

Tree1

$$\begin{array}{c}
 \rightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \\
 |(\rightarrow \Rightarrow) \\
 (a \Rightarrow b) \rightarrow (\neg b \Rightarrow a) \\
 |(\rightarrow \Rightarrow) \\
 \neg b, (a \Rightarrow b) \rightarrow a \\
 \wedge (\Rightarrow \rightarrow) \\
 \begin{array}{cc}
 \neg b \rightarrow a, a & \neg b, a \rightarrow a \\
 |(\neg \rightarrow) & |(\neg \rightarrow) \\
 \rightarrow b, a, a & a \rightarrow b, a \\
 \text{non - axiom} & \text{axiom}
 \end{array}
 \end{array}$$

Leaf $\rightarrow b, a, a$ is not an axiom

Tree2

$$\begin{array}{c}
 \rightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \\
 |(\rightarrow \Rightarrow) \\
 (a \Rightarrow b) \rightarrow (\neg b \Rightarrow a) \\
 \wedge (\Rightarrow \rightarrow) \\
 \begin{array}{cc}
 \rightarrow (\neg b \Rightarrow a), a & b \rightarrow (\neg b \Rightarrow a) \\
 |(\Rightarrow \rightarrow) & |(\Rightarrow \rightarrow) \\
 \neg b \rightarrow a, a & b, \neg b \rightarrow a \\
 |(\neg \rightarrow) & |(\neg \rightarrow) \\
 \rightarrow b, a, a & b \rightarrow b, a \\
 \text{non - axiom} & \text{axiom}
 \end{array}
 \end{array}$$

Leaf $\rightarrow b, a, a$ is not an axiom

Tree3

$$\begin{array}{c}
 \rightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \\
 |(\rightarrow \Rightarrow) \\
 (a \Rightarrow b) \rightarrow (\neg b \Rightarrow a) \\
 |(\rightarrow \Rightarrow) \\
 \neg b, (a \Rightarrow b) \rightarrow a \\
 |(\neg \rightarrow) \\
 (a \Rightarrow b) \rightarrow b, a \\
 \wedge (\Rightarrow \rightarrow) \\
 \begin{array}{cc}
 \rightarrow a, b, a & b \rightarrow b, a \\
 \text{non-axiom} & \text{axiom}
 \end{array}
 \end{array}$$

Leaf $\rightarrow b, a, a$ is not an axiom

All trees have at least one non-axiom leaf, thus, $\not\vdash_{GL} A$

2. We construct (by strong soundness of **GL**) a counter-model for A by making non-axiom leaf $\rightarrow b, a, a$ false. We assign $v(a)=F$, $v(b)=F$.