Extra Credit   Lecture 9
CSE541

Exercise 1: The formulas 1. - 9. That we assumed to be provable in S are those needed for 2 proofs of the Completeness Theorem. List the formulas that are needed for the Proof 1 only.

1. (A ⇒ (B ⇒ A))

2. ((A ⇒ (B ⇒ C)) ⇒ ((A ⇒ B) ⇒ (A ⇒ C)))

4. (A ⇒ A)

5. (B ⇒ ¬¬B)

6. (¬A ⇒ (A ⇒ B))

7. (A ⇒ (¬B ⇒ ¬(A ⇒ B)))

8. ((A ⇒ B) ⇒ ((¬A ⇒ B) ⇒ B))

1: (B1 ⇒ (A ⇒ B1)) is used in the base step in the proof for deduction theorem and in the inductive step of the case A is (A1 ⇒ A2) in the proof for the main lemma.

2: ((A ⇒ (B ⇒ C)) ⇒ ((A ⇒ B) ⇒ (A ⇒ C))) is used in the case 2 of the inductive step in the proof for the deduction theorem.

4. (A ⇒ A) is used in the case 2 of the base step in the proof for the deduction theorem.

5. (B ⇒ ¬¬B) is used in the inductive step of the case A is ¬A1 in the proof for the main lemma.

6. (¬A ⇒ (A ⇒ B)) is used in the inductive step of the case A is (A1 ⇒ A2) in the proof for the main lemma too.

7. (A ⇒ (¬B ⇒ ¬(A ⇒ B))) is used in the inductive step of the case A is (A1 ⇒ A2) in the proof for the main lemma.

8. ((A ⇒ B) ⇒ ((¬A ⇒ B) ⇒ B)) is used in the proof 1 for the completeness theorem along with Main Lemma , the Deduction Theorem, monotonicity, and Modus Ponens.

Exercise 2: We proved Completeness Theorem for the language L{⇒,¬}. Extend this proof to the language L{⇒,∪,¬} by adding all new CASES and needed PROVABLE formulas to our list 1. - 9. or to a shorter list from solution of the Exercise 1.

We define the system S as follows S = (L{⇒,∪,¬}, F , LA, (MP)) where the set of logical axioms LA ⊆ T is such that the formulas listed below are provable in S

1. (A ⇒ (B ⇒ A))

2. ((A ⇒ (B ⇒ C)) ⇒ ((A ⇒ B) ⇒ (A ⇒ C)))

3. ((¬B ⇒ ¬A) ⇒ ((¬B ⇒ A) ⇒ B))

4. (A ⇒ A)

5. (B ⇒ ¬¬B)

6. (¬A ⇒ (A ⇒ B))

7. (A ⇒ (¬B ⇒ ¬(A ⇒ B)))

8. ((A ⇒ B) ⇒ ((¬A ⇒ B) ⇒ B))

9. ((¬A ⇒ A) ⇒ A)

10. (A ⇒ (A∪B))

11. (¬A ⇒ (¬B ⇒ ¬ (A∪B)))

Proof of (¬A ⇒ (¬B ⇒ ¬ (A∪B))) for Tautology:

Assume (¬A ⇒ (¬B ⇒ ¬ (A∪B))) is not a tautology, meaning it has a counter model v. (use shorthand notation)

(¬A ⇒ (¬B ⇒ ¬ (A∪B))) = F if and only if

¬A = T and (¬B ⇒ ¬ (A∪B)) = F if and only if

¬B = T and ¬ (A∪B) = F if and only if

A = T or B = T, which contradicts with ¬A = T and ¬B = T. Therefore, such a v does not exist, and (¬A ⇒ (¬B ⇒ ¬ (A∪B))) is a tautology.

Proof of the Main Lemma: adding the proof for the case A is (A1 ∪ A2).

If A is (A1 ∪ A2) then A1 and A2 have less than n connectives A = A(b1, ... bn) so there are some subsequences c1, ..., ck and d1, ...dm for k, m ≤ n of the sequence b1, ..., bn such that A1 = A1(c1, ..., ck) and A2 = A(d1, ...dm) proof of the Main Lemma A1 and A2 have less than n connectives and so by the inductive assumption we have appropriate formulas C1, ..., Ck and D1, ...Dm such that C1, C2, . . . , Ck |- A1' and D1, D2, . . . , Dm |- A2' and C1, C2, ..., Ck , D1, D2, ..., Dm are subsequences of formulas B1, B2, ..., Bn corresponding to the propositional variables in A By the inductive assumption and monotonicity we have B1, B2, ..., Bn |- A1' and B1, B2, ..., Bn |- A2'.

Case: v ∗ (A1) = T

If v ∗ (A1) = T then A1' = A1

Observe that if v ∗ (A1) = T then A1' is A1 and, whatever value v gives A2, we have v ∗ (A1 ∪ A2) = T So A' is (A1 ∪ A2)

By the above and the inductive assumption B1, B2, ..., Bn |- A1 and since we have assumed 10. about S and by monotonicity we have B1, B2, ..., Bn |- (A1 ⇒ (A1 ∪ A2)) By above and MP we have B1, B2, ..., Bn |- (A1 ∪ A2) that is B1, B2, ..., Bn |- A.

Case: $v * (A2) = T$ similar to $v * (A1) = T$.

Case: $v * (A1) = F$, $v * (A2) = F$

If $v * (A1) = F$ then $A1' = \neg A1$ and if $v * (A2) = F$ then $A2' = \neg A2$. Also we have in this case $v * (A1 \cup A2) = F$ and so $A' = \neg(A1 \cup A2)$

By the above, the inductive assumption and monotonicity $B1, B2, ..., Bn \mid\!- \neg A1$ and also $B1, B2, ..., Bn \mid\!- \neg A2$.

Since we have assumed 11. about S and by monotonicity we have $B1, B2, ..., Bn \mid\!- (\neg A1 \Rightarrow (\neg A2 \Rightarrow \neg(A1 \cup A2)))$ By above and MP twice we have $B1, B2, ..., Bn \mid\!- \neg(A1 \cup A2)$ that is $B1, B2, ..., Bn \mid\!- A'$.

Now, the extended language has the Proof of Completeness Theorem as in the original language because of monotonicity.