CSCE 420 - Fall 2025

Homework 3 (HW3)

Ayushi Sharma

1a. Prove that (A^B→C^D) |- (A^B→C) ("conjunctive rule splitting") is a ***sound rule-of-inference*** *using a* ***truth table****.*

A∧B	**C**	**D**	**C∧D**	**A∧B→C∧D (Premise)**	**A∧B→C (Conclusion)**
T	T	T	T	T	T
T	T	F	F	F	T
T	F	T	F	F	F
T	F	F	F	F	F

In every row in which the premise is true (first row), the conclusion is also true. So, (A∧B→C∧D) entails (A∧B→C) and **the rule is sound**.

1b. Also prove (A^B→C^D) |= (A^B→C) using ***Natural Deduction.***

(Hint: it might help to use a ROI for "Implication Introduction". If you have a Horn clause, with 1 positive literal and *n-1* negative literals, like (¬X∨Z∨¬Y), you can transform it into a conjunctive rule by collecting the negative literals as positive antecedents, e.g. X^Y→Z. This is a truth-preserving operation (hence sound), which you could prove to yourself using a truth table.)

Natural Deduction Proof:

a. A ∧ B → C ∧ D (premise)
b. Assume A ∧ B (assumption for →-intro)
c. C ∧ D (1,2, →-elim)
d. C (3, ∧-elim)
e. A ∧ B → C (2–4, →-intro)

1c. Also prove (A^B→C^D) |= (A^B→C) using **Resolution**.

CNF:

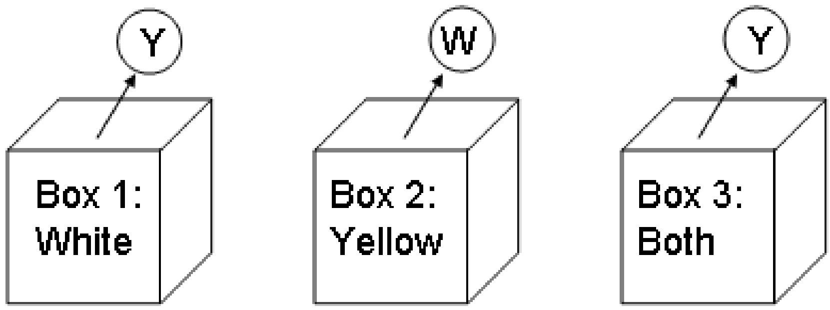
- Premise $A \land B \rightarrow C \land D \equiv (\neg A \lor \neg B \lor C) \land (\neg A \lor \neg B \lor D)$
- Negate goal: $\neg(A \land B \rightarrow C) \equiv A \land B \land \neg C$ (clauses A, B, ¬C)

Resolve:

- $(\neg A \lor \neg B \lor C)(\neg A \lor \neg B \lor C)$ with $\neg C \Rightarrow (\neg A \lor \neg B$
- With $A \Rightarrow \neg B$
- With B ⇒ contradiction. So, $A \land B \rightarrow C$ follows.

2. <u>**Sammy's Sport Shop**</u>

You are the proprietor of *Sammy's Sport Shop*. You have just received a shipment of three boxes filled with tennis balls. One box contains only yellow tennis balls, one box contains only white tennis balls, and one contains both yellow and white tennis balls. You would like to stock the tennis balls in appropriate places on your shelves. Unfortunately, the boxes have been labeled incorrectly; the manufacturer tells you that you have exactly one box of each, but that **each box is definitely labeled wrong**. You draw one ball from each box and observe its color. Given the initial (incorrect) labeling of the boxes above, and the three observations, use Propositional Logic to infer the correct contents of the middle box.



Use propositional symbols in the following form: O1Y means a yellow ball was drawn (observed) from box 1, L1W means box 1 was initially labeled white, C1W means box 1 contains (only) white balls, and C1B means box 1 actually contains both types of tennis balls. Note, there is no 'O1B', etc, because you can't directly "observe both". When you draw a tennis ball, it will either be white or yellow.

The initial facts describing this particular situation are: {O1Y, L1W, O2W, L2Y, O3Y, L3B}

2a. Using these propositional symbols, <u>write a propositional knowledge base</u> (sammy.kb) that captures the knowledge in this domain (i.e. implications of what different observations or labels mean, as well as constraints inherent in this problem, such as that all boxes have different contents). *Do it in a complete and general way*, writing down *all* the rules and constraints, not just the ones needed to make the specific inference about the middle box. *Do not include derived knowledge* that depends on the particular labeling of this instance shown above; stick to what is stated in the problem description above. Your KB should be general enough to reason about any alternative scenario, not just the one given above (e.g. with different observations and labels and box contents).

1. Each box can only have one type of content: For every box *i* in {1, 2, 3}:

- It must contain exactly one type of ball — yellow, white, or both:
  CiY ∨ CiW ∨ CiB
- And it can't have more than one type at the same time:
  ¬(CiY∧CiW), ¬(CiY∧CiB), ¬(CiW∧CiB)

Each ball type appears in only one box:

For each type *T* ∈ {Y, W, B}:

- There's at least one box with that type:
  $C1T \lor C2T \lor C3T$
- But no two boxes share the same type:
  For $i \neq j$: $\neg(CiT \land CjT)$

Observations tell us something about contents:

- If I pull out a yellow ball, that box must be either all yellow or mixed:
  $OiY \rightarrow (CiY \lor CiB)$
- If I pull out a white ball, that box must be either all white or mixed:
  $OiW \rightarrow (CiW \lor CiB)$

Every label is wrong:

- If a box is labeled yellow, it's not actually yellow:
  $LiY \rightarrow \neg CiY$
- If labeled white, it's not actually white:
  $LiW \rightarrow \neg CiW$
- If labeled both, it's not actually both:
  $LiB \rightarrow \neg CiB$

Given instance facts:
{O1Y, L1W, O2W, L2Y, O3Y, L3B}

2b. Prove that box 2 must contain white balls (**C2W**) using **Natural Deduction**.

1. From **L3B** and the rule that all labels are wrong:
   $\rightarrow \neg C3B$
2. From **O3Y**:
   $\rightarrow C3Y \lor C3B$
3. Combining these gives us:
   $\rightarrow C3Y$
4. Since there's only one yellow box, this means:
   $\rightarrow \neg C1Y$ and $\neg C2Y$
5. From **L1W** and the labels wrong rule:
   $\rightarrow \neg C1W$
6. Now, with $\neg C1Y$ and $\neg C1W$, the only option remaining for Box 1 is:
   $\rightarrow C1B$
7. Because there's only one "both" box here:
   $\rightarrow \neg C2B$
8. From **O2W**, we know:
   $\rightarrow C2W \lor C2B$
9. Using $\neg C2B$ from above, we can conclude:
   $\rightarrow C2W$
10. So, **Box 2 must contain white balls.**

2c. Convert your KB to CNF.

For each *i*:

- (CiY ∨ CiW ∨ CiB)
- (¬CiY ∨ ¬CiW), (¬CiY ∨ ¬CiB), (¬CiW ∨ ¬CiB)

For each type T:

- (C1T ∨ C2T ∨ C3T); and for i≠j: (¬CiT ∨ ¬CjT)

Observations:

- (¬OiY ∨ CiY ∨ CiB), (¬OiW ∨ CiW ∨ CiB)

Labels wrong:

- (¬LiY ∨ ¬CiY), (¬LiW ∨ ¬CiW), (¬LiB ∨ ¬CiB)

Instance facts as unit clauses: {O1Y, L1W, O2W, L2Y, O3Y, L3B}

2d. Prove C2W using **Resolution.**

We'll assume ¬C2W and look for a contradiction.

1. From **L3B**, and knowing labels are wrong:
   → ¬C3B
2. From **O3Y**, we have:
   → C3Y ∨ C3B
3. Together, that simplifies to:
   → C3Y
4. By uniqueness of yellow boxes:
   → ¬C1Y and ¬C2Y
5. From **L1W**, and the "labels wrong" rule:
   → ¬C1W
6. With ¬C1Y and ¬C1W, Box 1 must then be:
   → C1B
7. Because only one box can be "both":
   → ¬C2B
8. From **O2W**, we know:
   → C2W ∨ C2B
9. But since ¬C2B, this forces:
   → C2W

This contradicts our assumption of ¬C2W.

So, C2W is entailed: **Box 2 contains white balls.**

3. Do **Forward Chaining** for the *CanGetToWork* KB below.

You don't need to follow the formal FC algorithm (with agenda/queue and counts array). Just indicate which rules are triggered (in any order), and keep going until all consequences are generated.
Show the final list of all inferred propositions at the end. *Is CanGetToWork among them?*

```
KB = { a. CanBikeToWork → CanGetToWork
       b. CanDriveToWork → CanGetToWork
       c. CanWalkToWork → CanGetToWork
       d. HaveBike ∧ WorkCloseToHome ^ Sunny → CanBikeToWork
       e. HaveMountainBike → HaveBike
       f. HaveTenSpeed → HaveBike
       g. OwnCar → CanDriveToWork
       h. OwnCar → MustGetAnnualInspection
       i. OwnCar → MustHaveValidLicense
       j. CanRentCar → CanDriveToWork
       k. HaveMoney ∧ CarRentalOpen → CanRentCar
       l. HertzOpen→ CarRentalOpen
       m. AvisOpen→ CarRentalOpen
       n. EnterpriseOpen→ CarRentalOpen
       o. CarRentalOpen → IsNotAHoliday
       p. HaveMoney ∧ TaxiAvailable → CanDriveToWork
       q. Sunny ^ WorkCloseToHome → CanWalkToWork
       r. HaveUmbrella ^ WorkCloseToHome → CanWalkToWork
       s. Sunny → StreetsDry }
```

Facts: { `Rainy, HaveMoutainBike, EnjoyPlayingSoccer, WorkForUniversity, WorkCloseToHome, HaveMoney, HertzClosed, AvisOpen, McDonaldsOpen` }

Rules that fire (in logical order):

e. HaveMountainBike → HaveBike
→ From this, we can infer **HaveBike**

m. AvisOpen → CarRentalOpen
→ Since AvisOpen is true, infer **CarRentalOpen**

o. CarRentalOpen → IsNotAHoliday
→ Infer **IsNotAHoliday**

k. HaveMoney ∧ CarRentalOpen → CanRentCar
→ Both conditions are true, so infer **CanRentCar**

j. CanRentCar → CanDriveToWork
→ Infer **CanDriveToWork**

b. CanDriveToWork → CanGetToWork
→ Infer **CanGetToWork**

New inferred propositions:
{ HaveBike, CarRentalOpen, IsNotAHoliday, CanRentCar, CanDriveToWork, CanGetToWork }

Conclusion:
Yes, CanGetToWork is among the inferred propositions.

4. Do **Backward Chaining** for the *CanGetToWork* KB.

In this case, you should follow the BC algorithm closely (the pseudocode for the propositional version of Back-chaining is given in the lecture slides).

Important: when you pop a subgoal (proposition) from the goal stack, you should systematically go through all rules that can be used to prove it IN THE ORDER THEY APPEAR IN THE KB. In some cases, this will lead to *back-tracking*, which you should show.

Also, the sequence of results depends on order in which antecedents are pushed onto the stack. If you have a rule like A^B→C, and you pop C off the stack, push the antecedents in reverse order, so B goes in first, then A; in the next iteration, A would be the next subgoal popped off the stack.

Goal stack start: ***CanGetToWork***

Step 1: Prove *CanGetToWork*

Rules:
a) *CanBikeToWork → CanGetToWork*
b) *CanDriveToWork → CanGetToWork*
c) *CanWalkToWork → CanGetToWork*

Try a. Bike path

Subgoal: ***CanBikeToWork***.
Rules with consequent **CanBikeToWork** is only **d** (*HaveBike ∧ WorkCloseToHome ∧ Sunny → CanBikeToWork*).

- Use **d**, (original: HaveBike, WorkCloseToHome, Sunny):
  Goal stack becomes: (**Sunny**, **WorkCloseToHome**, **HaveBike**)
- Pop **HaveBike**:
  Rules that conclude HaveBike: **e** (HaveMountainBike → HaveBike), **f** (HaveTenSpeed → HaveBike).
  • Try **e** first (by order): need **HaveMountainBike**. Since this is a given fact, **HaveBike** succeeds.
- Pop **WorkCloseToHome**: given fact, **succeeds!!**
- Pop **Sunny**: not a fact; no rule concludes Sunny, so **Sunny fails**.
  → Rule **d** fails, and no other rule yields CanBikeToWork ⇒ **bike branch fails**.

Try b. Drive path

Subgoal: ***CanDriveToWork***.
Rules with consequent **CanDriveToWork**, in order:
g) *OwnCar → CanDriveToWork*

j) *CanRentCar → CanDriveToWork*
p) *HaveMoney ∧ TaxiAvailable → CanDriveToWork*

- Try **g**: needs **OwnCar** (not a fact, no rule derives it) ⇒ **fail**
- Try **j**: needs **CanRentCar**. Push **CanRentCar**.
  Subgoal: ***CanRentCar***.
  Rules with consequent **CanRentCar**: **k** (HaveMoney ∧ CarRentalOpen → CanRentCar).
  • Use **k**; (original: HaveMoney, CarRentalOpen):
  Goal stack becomes: (**CarRentalOpen**, **HaveMoney**)
    - Pop **HaveMoney**: given fact, **succeeds**!
    - Pop **CarRentalOpen**:
      Rules that conclude CarRentalOpen (in order): **l** (HertzOpen → CarRentalOpen), **m** (AvisOpen → CarRentalOpen), **n** (EnterpriseOpen → CarRentalOpen).
      • **l** needs HertzOpen (we actually have **HertzClosed**) ⇒ fail.
      • **m** needs **AvisOpen** — **given fact**, so infer **CarRentalOpen**.
      • So, **CanRentCar** succeeds by **k**.
    - With **CanRentCar**, rule **j** gives **CanDriveToWork** (**succeeds**!)
- With **CanDriveToWork**, rule **b** gives **CanGetToWork**. **Success!!**

Final answer: The derivation completes and succeeds via the **drive (rent)** route, using **m→k→j→b** in order, with backtracking from the failed bike path (missing Sunny) and failed own car path.