Exercises Lecture 6
Intelligent Systems Programming (ISP)

Exercise 1

Consider the claims

1. If the sun shines then it is true that I get wet if it rains.
2. If the sun shines then it is summer.
3. It is not summer
4. Therefore, I get wet if the sun shines.

a) A proposition symbol represents something that can be either true or false. In claim 1. , 2., 3. and 4. above, there are several statements like "the sun shines" that can be either true or false. Define a propositional symbol for each of these statements.
b) Translate 1., 2., 3. , and 4. to propositional logic sentences using your proposition symbols from a).
c) Does the conjunction of 1., 2. and 3. entail the conclusion in 4 (why / why not)?

Exercise 2 (adapted from RN03 7.8)

Decide whether each of the following sentences is valid, unsatisfiable, or neither. Verify your decisions using truth tables or equivalence rules.

a) $Smoke \Rightarrow Smoke$
b) $Smoke \Rightarrow Fire$
c) $(Smoke \Rightarrow Fire) \Rightarrow (\neg Smoke \Rightarrow \neg Fire)$
d) $Smoke \vee Fire \vee \neg Fire$
e) $((Smoke \wedge Heat) \Rightarrow Fire) \Leftrightarrow ((Smoke \Rightarrow Fire) \vee (Heat \Rightarrow Fire))$
f) $(Smoke \Rightarrow Fire) \Rightarrow ((Smoke \wedge Heat) \Rightarrow Fire)$
g) $Big \vee Dumb \vee (Big \Rightarrow Dumb)$
h) $(Big \wedge Dumb) \vee \neg Dumb$

Exercise 3 (adapted from RN03 7.4)

Argue for the correctness of each of the following assertions:

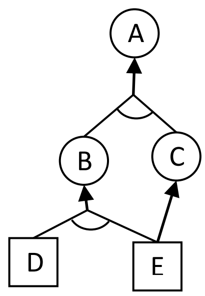
a) α is valid if and only if $True \vDash \alpha$.
b) For any α, $False \vDash \alpha$.
c) $\alpha \vDash \beta$ if and only if the sentence $(\alpha \Rightarrow \beta)$ is valid.
d) $\alpha \equiv \beta$ if and only if the sentence $(\alpha \Leftrightarrow \beta)$ is valid.
e) $\alpha \vDash \beta$ if and only if the sentence $(\alpha \wedge \neg \beta)$ is unsatisfiable.

Exercise 4

a) Translate the sentence $\neg(a \Rightarrow b) \wedge (a \vee b \Leftrightarrow (c \Rightarrow b))$ to CNF
b) Use resolution to prove $\neg(a \Rightarrow b) \wedge (a \vee b \Leftrightarrow (c \Rightarrow b)) \models \neg c$

Mandatory assignment

During the lecture 4 you learned that the forward-chaining algorithm can be used to efficiently entail propositions from a knowledge-base of horn-clauses. In RN13 p. 231, a different approach called backward-chaining is introduced. This algorithm works backward from the query. If the query is known to be true, then no work is needed, otherwise it tries to prove the query by proving its premises true (by backward-chaining). Consider the following AND-OR graph of a knowledge-base *KB*:



If we were to query the *KB* for A using backward-chaining, we would first have to prove B and C. C can be proven because it is implied by E which is a known fact, and B can be proven because it is implied by D and E which are known facts. Since we have proven B and C we can conclude A.

1) In which situations can it be an advantage to do backward-chaining rather than forward-chaining?
2) Professor Smart has implemented the following backward-chaining algorithm:

function PL-BC-ENTAILS(*KB*, *q*) **returns** *true* or *false*
  inputs: *KB*, the knowledge base, a set of propositional definite clauses
    q, the query, a propositional symbol
  if *q* is known to be true in *KB* **then return** *true*
  for each clause *c* in *KB* where *q* is in c.CONCLUSION **do**
    if CHECK-ALL(*KB*, *c*.PREMISE) **then return** *true*
  return *false*

function CHECK-ALL(*KB*, *premise*) **returns** *true* or *false*
  inputs: *KB*, the knowledge base, a set of propositional definite clauses
    premise, a set of propositional symbols
  for each *p* in *premise* **do**
    if not PL-BC-Entails(*KB*, *p*) **then return** *false*
  return *true*

Given the following *KB* of horn-clauses:

- $F \wedge E \Rightarrow D$
- $E \Rightarrow B$
- $B \wedge E \wedge G \Rightarrow C$
- $C \Rightarrow G$
- $D \wedge B \wedge C \Rightarrow A$
- $\Rightarrow F$
- $\Rightarrow E$

Can professor Smart's algorithm answer$KB \vDash A$? If yes, explain how the algorithm induces the answer. If no, explain in words where the problem lies and extend professor Smart's algorithm with a solution.

3) The book mentions that an efficient implementation of a backward-chaining algorithm runs in linear time. Does professor Smart's algorithm run in linear time? If yes, explain why. If no, explain why not and extend professor Smart's algorithm with improvements that reduces its runtime. It is not required that you come up with a linear time algorithm.