Theory of Computer Science

G. Röger
Spring Term 2020

University of Basel
Computer Science

Exercise Sheet 2 — Solutions

Exercise 2.1 (Equivalences; 1.5+1.5 marks)

(a) Transform the following formula into CNF by applying the equivalence rules shown in the lecture. For each step, only apply one equivalence rule and also specify it.

$$\phi = (\neg(A \leftrightarrow \neg B) \rightarrow C)$$

Solution:

$$\begin{aligned}
\phi &= (\neg(A \leftrightarrow \neg B) \rightarrow C) & \\
&\equiv (\neg\neg(A \leftrightarrow \neg B) \lor C) & (\rightarrow\text{-Elimination})\\
&\equiv ((A \leftrightarrow \neg B) \lor C) & (\text{Double negation})\\
&\equiv (((A \rightarrow \neg B) \land (\neg B \rightarrow A)) \lor C) & (\leftrightarrow\text{-Elimination})\\
&\equiv (((\neg A \lor \neg B) \land (\neg B \rightarrow A)) \lor C) & (\rightarrow\text{-Elimination})\\
&\equiv (((\neg A \lor \neg B) \land (\neg\neg B \lor A)) \lor C) & (\rightarrow\text{-Elimination})\\
&\equiv (((\neg A \lor \neg B) \land (B \lor A)) \lor C) & (\text{Double negation})\\
&\equiv (C \lor ((\neg A \lor \neg B) \land (B \lor A))) & (\text{Commutativity})\\
&\equiv ((C \lor (\neg A \lor \neg B)) \land (C \lor (B \lor A))) & (\text{Distributivity})
\end{aligned}$$

(b) Prove that the following formula is a tautology by showing that $\phi \equiv (A \lor \neg A)$ holds. Use the equivalence rules from the lecture, only apply one rule for each step and specify the applied rule.

$$\phi = (A \lor (\neg(A \land \neg(\neg A \land C)) \lor (A \land B)))$$

Solution:

$$\begin{aligned}
\phi &= (A \lor (\neg(A \land \neg(\neg A \land C)) \lor (A \land B))) & \\
&\equiv (A \lor ((A \land B) \lor \neg(A \land \neg(\neg A \land C)))) & (\text{Commutativity})\\
&\equiv ((A \lor (A \land B)) \lor \neg(A \land \neg(\neg A \land C))) & (\text{Associativity})\\
&\equiv (A \lor \neg(A \land \neg(\neg A \land C))) & (\text{Absorption})\\
&\equiv (A \lor (\neg A \lor \neg\neg(\neg A \land C))) & (\text{De Morgan})\\
&\equiv (A \lor (\neg A \lor (\neg A \land C))) & (\text{Double negation})\\
&\equiv (A \lor \neg A) & (\text{Absorption})
\end{aligned}$$

Exercise 2.2 (Inference; 1+1+1 marks + 1 bonus mark)

You'll find a Java program on the lecture website that checks proofs formulated in propositional logic. Use this program to prove the following statements. For a statement of the form WB $\models \varphi$ write a text file containing a derivation that only uses formulas from WB as assumptions and that has φ in its last line. An example for this is contained in the file `proof.txt`.
The program checks WB $\vdash \varphi$. Since the proof system used by the program is correct, this implies WB $\models \varphi$.

(a) $\{A, B\} \models ((A \land B) \lor C)$

Solution:

```
A                    | Assumption   |
B                    | Assumption   |
(A /\ B)             | AndIntro     | 1, 2
((A /\ B) \/ C)      | OrIntroLeft  | 3
```

(b) $\{(A \land B)\} \models (A \to (B \lor C))$

Solution:

```
(A /\ B)             | Assumption         |
B                    | AndElimLeft        | 1
(B \/ C)             | OrIntroLeft        | 2
(~A \/ (B \/ C))     | OrIntroRight       | 3
(A -> (B \/ C))      | ImplicationIntro   | 4
```

(c) $\{((A \lor B) \to (A \to C)), A\} \models C$

Solution:

```
A                          | Assumption    |
(A \/ B)                   | OrIntroLeft   | 1
((A \/ B) -> (A -> C))     | Assumption    |
(A -> C)                   | ModusPonens   | 2, 3
C                          | ModusPonens   | 1, 4
```

(d) $\{((C \lor D) \leftrightarrow (A \land B)), \neg E, (((A \land B) \land (C \lor D)) \to E)\} \models \neg(A \land B)$
For this exercise, extend the calculus by a new rule *negation-introduction*:

$$\frac{(\varphi \to \psi), (\varphi \to \neg\psi)}{\neg\varphi}$$

Solution:

```
((C \/ D) <-> (A /\ B))           | Assumption              |
((A /\ B) -> (C \/ D))            | BiimplicationElimLeft   | 1
~E                                | Assumption              |
(((A /\ B) /\ (C \/ D)) -> E)     | Assumption              |
~((A /\ B) /\ (C \/ D))           | ModusTollens            | 3, 4
(~(A /\ B) \/ ~(C \/ D))          | DeMorgan2LeftToRight    | 5
((A /\ B) -> ~(C \/ D))           | ImplicationIntro        | 6
~(A /\ B)                         | NegationIntro           | 2, 7
```

The additional rule has to be added to `Calculus.java` with the following line:

```
addRule("NegationIntro", "(X -> Y), (X -> ~Y) |- ~X");
```

(e) *Bonus exercise:* To show that a calculus is correct, we have to prove that all rules are correct. Show the correctness of the rule *negation-introduction*

Solution:

Let $\mathcal{I}$ be a model of $(\varphi \rightarrow \psi)$ and $(\varphi \rightarrow \neg\psi)$. From $\mathcal{I} \models (\varphi \rightarrow \psi)$ we know that $\mathcal{I} \not\models \varphi$ or $\mathcal{I} \models \psi$ must be true. If we assume $\mathcal{I} \models \varphi$ then $\mathcal{I} \models \psi$ must be true (*). But from $\mathcal{I} \models (\varphi \rightarrow \neg\psi)$ we know that $\mathcal{I} \not\models \varphi$ or $\mathcal{I} \models \neg\psi$ must hold. Since $\mathcal{I} \models \varphi$ is true according to our assumption, $\mathcal{I} \models \neg\psi$ must be true. This is a contradiction to (*). The assumption $(\mathcal{I} \models \varphi)$ must be wrong and we have $\mathcal{I} \models \neg\varphi$.

Since all models $\mathcal{I}$ of $(\varphi \rightarrow \psi)$ and $(\varphi \rightarrow \neg\psi)$ are also models of $\mathcal{I} \models \neg\varphi$, we conclude that the rule is correct: $\{(\varphi \rightarrow \psi), (\varphi \rightarrow \neg\psi)\} \models \neg\varphi$.

Note on the submission process: please create one text file for each exercise part which contains the derivation. The program must be able to parse the file and accept the derivation as correct. The new rule (*negation-introduction*) requires a new line in the program. Copy this line on your regular submission. The bonus exercise cannot be solved with the program.

Exercise 2.3 (Refutation Theorem; 2 marks)
Prove the refutation theorem, that is, show for any set of formulas KB and any formula φ that

$$\mathrm{KB} \cup \{\varphi\} \text{ is unsatisfiable if and only if } \mathrm{KB} \models \neg\varphi.$$

Solution:
"$\Rightarrow$": If $\mathrm{KB} \cup \{\varphi\}$ is unsatisfiable then there is no interpretation $\mathcal{I}$ with $\mathcal{I} \models \mathrm{KB}$ and $\mathcal{I} \models \varphi$. Hence for every $\mathcal{I}$ with $\mathcal{I} \models \mathrm{KB}$ it holds that $\mathcal{I} \not\models \varphi$ and we conclude that $\mathrm{KB} \models \neg\varphi$.

"$\Leftarrow$": If $\mathrm{KB} \models \neg\varphi$ then it holds for all $\mathcal{I}$ with $\mathcal{I} \models \mathrm{KB}$ that $\mathcal{I} \models \neg\varphi$ and hence $\mathcal{I} \not\models \varphi$. Therefore there is no interpretation with $\mathcal{I} \models \mathrm{KB} \cup \{\varphi\}$, so $\mathrm{KB} \cup \{\varphi\}$ is unsatisfiable.

Exercise 2.4 (Refutation Completeness; 2 marks)
Let P be a computer program that takes a set of propositional logic formulas as input and returns whether this set of formulas is unsatisfiable.
How can you use P to decide for a knowledge base KB and a propositional logic formula φ whether

(a) KB is satisfiable?

Solution:

We run the program for input KB. Then KB is satisfiable iff the program returns that the input is not unsatisfiable.

(b) $\mathrm{KB} \models \varphi$?

Solution:

We run the program on input $\mathrm{KB} \cup \{\neg\varphi\}$. According to the refutation theorem it holds that $\mathrm{KB} \models \varphi$ iff the program returns that the input is unsatisfiable.

(c) KB is a tautology?

Solution:

A knowledge base KB is a tautology iff every interpretation is a model of every formula $\varphi \in \mathrm{KB}$ and therefore also a model of the conjunction $(\bigwedge_{\varphi \in \mathrm{KB}} \varphi)$. Hence, KB is a tautology iff there is no interpretation $\mathcal{I}$ such that $\mathcal{I} \not\models (\bigwedge_{\varphi \in \mathrm{KB}} \varphi)$. This means KB is a tautology iff $\neg(\bigwedge_{\varphi \in \mathrm{KB}} \varphi)$ is unsatisfiable. Thus KB is a tautology iff the program returns that input $\{\neg(\bigwedge_{\varphi \in \mathrm{KB}} \varphi)\}$ is not unsatisfiable.

(d) KB is falsifiable?

Solution:

A knowledge base is falsifiable iff it is not valid. Whether it is valid can be decided as described in part c.