# Final Exam 2ITX0 Applied Logic

January 21, 2021, 9:00 - 12:00

This examination consists of 8 problems for which the indicated number of points can be obtained. The grade is the obtained number of points divided by 10. **You must explain all your answers.**

### Problem 1.

(10 points)

Organize 64 boolean variables on a chess board as follows:

$$\begin{array}{cccc} p_{11} & p_{12} & \cdots & p_{18} \\ p_{21} & p_{22} & \cdots & p_{28} \\ \vdots & \vdots & & \vdots \\ p_{81} & p_{82} & \cdots & p_{88} \end{array}$$

Present a formula in proposition logic on these boolean variables in logic notation, expressing that in every column at least one value is true and in every row at most one value is false. Here you may use standard notations like $\bigwedge_{i=1}^{n} \cdots$.

### Problem 2.

(15 points) Consider the CNF consisting of the following eight clauses

$$\begin{array}{llll} (1) & \neg p \vee t & (5) & p \vee \neg r \vee s \\ (2) & \neg q \vee r & (6) & \neg q \vee s \vee \neg t \\ (3) & p \vee s & (7) & r \vee \neg s \vee \neg t \\ (4) & \neg r \vee \neg s & (8) & \neg p \vee q \end{array}$$

Establish whether this CNF is satisfiable by DPLL; start by case analysis on $p$. Indicate for every step the number of the clause that is used. If the formula is satisfiable, give the resulting satisfying assignment.

### Problem 3.

(10 points) Give the Tseitin transformation of the formula

$$(p \vee q) \to (\neg r \vee (p \wedge \neg s));$$

make clear what are the names of the subformulas that you introduce.

## Problem 4.

Consider the following memoryless information source $M$ (motion).

| message | *north* | *east* | *south* | *west* | *halted* |
|---|---|---|---|---|---|
| probability | 0.3 | 0.1 | 0.3 | 0.1 | 0.2 |

The following table shows an encoding $E_1$ of source $M$.

| message | encoding |
|---|---|
| *north* | 000 |
| *east* | 001 |
| *south* | 010 |
| *west* | 011 |
| *halted* | 1 |

**(a)** (1 point) How many bits per message are needed in a *fixed-length* encoding of this information source?

**(b)** (2 points) Using $E_1$, you receive 01011011100 . . . (the remaining bits are not yet available). What is the information you can extract from the received bits about the message sequence that was sent?

**(c)** (2 points) Encoding $E_1$ is not a fixed-length encoding. Why can every sequence of encoded messages be uniquely decoded anyways?

**(d)** (2 points) How many bits per message does encoding $E_1$ use *on average* when encoding source $M$?

**(e)** (4 points) What would the average number of bits per message be under a Huffman encoding? Show how you obtained this encoding (clearly indicate the steps) and its average number of bits per message.

**(f)** (2 points) Estimate the least number of bits per message that a lossless encoding needs, *on average*. Motivate your answer.

The following binary logarithms are given:

| $p$ | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
|---|---|---|---|---|---|---|---|---|---|
| $-\log_2 p$ | 3.32 | 2.32 | 1.74 | 1.32 | 1.00 | 0.737 | 0.515 | 0.322 | 0.152 |

## Problem 5.

We want to transmit $k$-bit messages, for some fixed integer $k \geq 1$, over a (noisy) binary symmetric channel. Consider message $s = s_1 s_2 \dots s_k$ and define (parity bit) $p = (s_1 + s_2 + \cdots + s_k) \bmod 2$; that is $p = 0$ if $s$ has an even number of 1's, and $p = 1$ if $s$ has an odd number of 1's. We now encode message $s$ as $sssppp$; that is, three copies of $s$ followed by two copies of $p$.

**(a)** (1 point) Write down the four codewords for $k = 2$

**(b)** (1 point) What is the rate of this encoding for $k = 2$?

**(c)** (1 point) What is the rate of this encoding in general, as function of $k$?

**(d)** (1 point) For $k = 2$, you receive the following 16 bits (spaces are for readability only)

11 11 10 11  11 10 11 10

What is the most likely message that was sent?

**(e)** (4 points) What is the smallest positive number $b_d$ of bit errors per message that is *not detectable* when using this encoding? Explain your answer. Also give a concrete example where $b_d$ bit errors are not detectable.

**(f)** (4 points) What is the smallest positive number $b_c$ of bit errors per message that is *not correctable with certainty* when using this encoding? Explain your answer. Also give a concrete example where $b_c$ bit errors are not correctable with certainty.

## Problem 6.

You have a 64-bit security key $b_1 b_2 \dots b_{64}$ for your password file. As a backup, you want to distribute the information in that key among four friends. Each friend receives a so-called 'share' in the form of a bit sequence, such that:

- From the shares of the four friends together, the security key can be reconstructed.
- With three or fewer of the shares, an enemy can do no better than trying all possible 64-bit keys, that is, no information about the security key can be obtained.

**(a)** (5 points) Explain quantitatively why giving friend $i$ ($1 \leq i \leq 4$) bits $b_{16(i-1)+1} b_{16(i-1)+2} \dots b_{16(i-1)+16}$ of the key is not a good idea.

**(b)** (5 points) Explain how to create shares securely, and why that works.

## Problem 7.

(10 points) Compute

$$wp(b := a;\ \text{if } a < b \text{ then } \langle c := a; b := a + c\rangle, a + b = c).$$

## Problem 8.

For a given value $b > 0$ we want to compute $b^2$ by only using addition, using the invariant $a = c^2$. The following incomplete Hoare triple is given:

$\{b > 0\}$
a := ? ; c := ? ;
while $c \neq b$ do
$\quad \langle a := a +$ ? ; $c := c + 1\rangle$
$\{a = b^2\}$

**(a)** (5 points)

Replace the three question marks '?' by expressions only using $0, 1, a, b, c, +$ such that the program is correct with respect to the given pre- and postcondition.

**(b)** (15 points)

Give the full proof (of partial correctness) using the given invariant and rules for *wp*.