Staff Solutions to Problem Set 2

Reading:

- Chapter 3.6. Predicate Formulas,
- Chapter 4. Mathematical Data Types,
- Chapter 5. Induction.

STAFF NOTE: Lectures covered: Predicate Formulas, Predicate Logic, Sets and Sequences, Binary Relations, and Induction

Problem 1. (a) Translate the following sentence into a predicate formula:

There is a student who has e-mailed at most \( n \) other people in the class, besides possibly himself.

The domain of discourse should be the set of students in the class; in addition, the only predicates that you may use are

- equality,
- \( E(x, y) \), meaning that “\( x \) has sent e-mail to \( y \).”

COMMENTS:

- PS\_e mailed\_at\_most\_n\_others
- from: S17.ps2
- small perturbation of PS\_e mailed\_at\_most\_2\_others

keywords = [ predicate formula translate sentence first-order logic ]

Solution. A good way to begin tackling this problem is by trying to translate parts of the sentence. First of all, our formula must be of the form

\[ \exists x. \text{atmost}_n(x) \]

where \( \text{atmost}_n(x) \) should be a formula that says that “student \( x \) has e-mailed at most \( n \) other people in the class, besides possibly himself”.

One way to express \( \text{atmost}_n(x) \) is “whenever we find a student \( s \) who has been e-mailed by \( x \), this student is either \( x \) or one of a particular set of students \( y_0, y_1, \ldots, y_{n-1} \)”.

A formula expressing that \( s \) is either \( x \) or one of students \( y_0, y_1, \ldots, y_{n-1} \) is

\[ s = x \text{ OR } s = y_0 \text{ OR } s = y_1 \text{ OR } \ldots \text{ OR } s = y_{n-1}. \]
So we can express “whenever we find a student \( s \) who has been e-mailed by \( x \ldots \)” with the formula

\[
\forall s. [E(x, s) \text{ IMPLIES } \ldots].
\]

Adding existential quantifiers to say that there are such students \( y_0, y_1, \ldots, y_{n-1} \), we finish with

\[
\forall s. [E(x, s) \text{ IMPLIES } (s = x \text{ OR } s = y_0 \text{ OR } s = y_1 \ldots \text{ OR } s = y_{n-1})].
\]

At this point you may be thinking that this formula actually says that “\( x \) has e-mailed exactly \( n \) students besides possibly himself.” But remember, the \( y_i \)’s may not all be different students. Of course we could have added that constraint with a long \text{ AND} \ formula:

\[
\begin{align*}
  y_0 & \neq y_1 \text{ AND } y_0 \neq y_2 \text{ AND } y_1 \neq y_3 \text{ AND } \ldots \text{ AND } y_0 \neq y_{n-1} \text{ AND } \\
  y_1 & \neq y_2 \text{ AND } y_1 \neq y_3 \text{ AND } \ldots \text{ AND } y_1 \neq y_{n-1} \text{ AND } \\
  & \vdots \\
  y_{n-2} & \neq y_{n-1}.
\end{align*}
\]

In the next part of this problem, we’ll see a simple expression that does this job and would be much shorter to write out.

\[\text{(b)}\] Explain how you would use your predicate formula (or some variant of it) to express the following two sentences.

1. There is a student who has emailed at least \( n \) other people in the class, besides possibly himself.
2. There is a student who has emailed exactly \( n \) other people in the class, besides possibly himself.

\textbf{Solution.} Student \( x \) has emailed at least \( n \) students just when he has \textit{not} emailed at most \( n - 1 \) students. So define

\[
\text{atleast}_n(x) ::= \text{NOT}(\text{atmost}_{n-1}(x)).
\]

and a formula for item 1. becomes

\[
\exists x. \text{atleast}_n(x).
\]

Now student \( x \) has emailed \textit{exactly} \( n \) students when he has emailed at most and also at least \( n \) students, so for item 2. we have

\[
\text{exactly}_n ::= \exists x. \text{atmost}_n(x) \text{ AND atleast}_n(x).
\]

\[\text{Problem 2.}\]

Let \( A, B \) and \( C \) be sets. Prove that

\[
A \cup B \cup C = (A - B) \cup (B - C) \cup (C - A) \cup (A \cap B \cap C)
\]

using a chain of \text{IFF}’s to reduce the set-theoretic equality to a propositional equivalence, as in the text.

\textbf{COMMENTS:}

- PS_set_union
- from: S17.ps2, F09.ps2, S06.ps1
Solution. Proof. We prove that an element $x$ is a member of the set described on the left hand side of equality (1) iff it is a member of the set described on the right hand side.

The key step in the proof uses the fact that the following two propositional formulas are equivalent.

\begin{align*}
P \lor Q \lor R, \quad \text{(2)} \\
(P \land \overline{Q}) \lor (Q \land \overline{R}) \lor (R \land \overline{P}) \lor (P \land Q \land R). \quad \text{(3)}
\end{align*}

There are multiple ways to verify this equivalence, and we will take it for granted.

$x \in A \cup B \cup C$
IFF $(x \in A) \lor (x \in B) \lor (x \in C)$ (by def of $\cup$)
IFF $((x \in A) \land \overline{x \in B})$ or
\begin{align*}
((x \in B) \land \overline{x \in C}) & \\
((x \in C) \land \overline{x \in A}) & \\
((x \in A) \land (x \in B) \land (x \in C)) & \quad \text{(equivalence of (2) and (3))}
\end{align*}
IFF $(x \in A - B) \lor (x \in B - C) \lor (x \in C - A)$
\begin{align*}
or (x \in A \cap B \cap C) & \quad \text{(by def of $- \land$)} \\
iff x \in (A - B) \cup (B - C) \cup (C - A) & \\
\quad \cup (A \cap B \cap C) & \quad \text{(by def of $\cup$)}
\end{align*}

Problem 3.
Let $R : A \to B$ and $S : B \to C$ be binary relations such that $S \circ R$ is a bijection and $|A| = 2$.

Give an example of such $R$, $S$ where neither $R$ nor $S$ is a function. Indicate exactly which properties—total, surjection, function, and injection—your examples of $R$ and $S$ have.

*Hint:* Let $|B| = 4$.

**COMMENTS:**

- FPjections_nofunc
- CH, S14
- mashup of smaller problems
- summary of properties part restored by ATRM 2/23/17

**keywords** = [ composition surjection injection bijection total function infinite ]

Solution. It is easy to see that $R$ must be total ([\geq 1 out]) and $S$ must be a surjection ([\geq 1 in]).

$C$ must have 2 elements since $A$ bij $C$.

For minimal example with $R$, $S$ not functions, see Figure 1. For an example in which $R$ is not even an injection, see Figure 2.

To explain: following the hint, we let $|B| = 4$. 
Figure 1  $S \circ R : A \rightarrow C$ is a bijection.

Figure 2  $S \circ R : A \rightarrow C$ is a bijection and $R$ is not an injection.
Figure 3  OR-circuit from AND-circuit.

We put an $R$-arrow from the first element of $A$ to the second element of $B$ and an $S$-arrow from this second element of $B$ to the first element of $C$; likewise from the second element of $A$ to the third element of $B$ and from this third element of $B$ to the second element of $C$. These arrows will provide the bijection defined by $S \circ R$.

We put two $R$-arrows into the first element in $B$ with no $S$-arrow out as in Figure 2; so $R$ is not a function or an injection and $S$ is not total. We put two $S$-arrows out of the fourth element of $B$ and no $R$-arrows in; so $R$ is not a surjection and $S$ is not a function or an injection.

So besides $R$ being total and $S$ a surjection, Figure 2 shows that neither $R$ nor $S$ need have any additional “jection” properties.

Problem 4.
A $k$-bit AND-circuit is a digital circuit that has $k$ 0-1 valued inputs $d_0, d_1, \ldots, d_{k-1}$ and one 0-1-valued output variable whose value will be

$$d_0 \text{ AND } d_1 \text{ AND } \cdots \text{ AND } d_{k-1}.$$

OR-circuits are defined in the same way, with “OR” replacing “AND.”

(a) Suppose we want an OR-circuit but only have a supply of AND-circuits and some NOT-gates (“inverters”) that have one 0-1 valued input and one 0-1 valued output. We can turn an AND-circuit into an OR-circuit by attaching a NOT-gate to each input of the AND-circuit and also attaching a NOT-gate to the output of the AND-circuit. This is illustrated in Figure 3. Briefly explain why this works.

COMMENTS:

- FP_AND_circuit_shorter_induction
- induction version of WOP problem FP_and_circuit_shorter
- overlaps FP_and_circuit
- revised by ARM to use from WOP to induction by ARM 2/19/17
- REVISE TO BACKREF FP_and_circuit_shorter and not repeat figures

keywords = [ circuit digital and tree ]

---

1Following the usual conventions for digital circuits, we’re using 1 for the truth value $T$ and 0 for $F$. 

Solution. Negating the inputs and the output of a $k$-bit AND-circuit with inputs $d_0, d_1, \ldots, d_{k-1}$ turns it into a circuit that whose output value will be

$$\text{NOT}(d_0 \text{ AND } d_1 \text{ AND } \cdots \text{ AND } d_{k-1})$$

which by DeMorgan’s Law is equivalent to

$$d_0 \text{ OR } d_1 \text{ OR } \cdots \text{ OR } d_{k-1}.$$ 

Large digital circuits are built by connecting together smaller digital circuits as components. One of the most basic components is a two-input/one-output AND-gate that produces an output value equal to the AND of its two input values. So according the definition in part (a), a single AND-gate is a 1-bit AND-circuit.

We can build up larger AND-circuits out of a collection of AND-gates in several ways. For example, one way to build a 4-bit AND-circuit is to connect three AND-gates as illustrated in Figure 4.

More generally, a depth-$n$ tree-design AND-circuit—“depth-$n$ circuit” for short—has $2^n$ inputs and is built from two depth-$(n-1)$ circuits by using the outputs of the two depth-$(n-1)$ circuits as inputs to a single AND-gate. This is illustrated in Figure 5. So the 4-bit AND-circuit in Figure 4 is a depth-2 circuit. A depth-1 circuit is defined simply to be a single AND-gate.

(b) Let gate#($n$) be the number of AND-gates in a depth-$n$ circuit. Prove by induction that

$$\text{gate#}(n) = 2^n - 1$$

for all $n \geq 1$. 

Figure 4 A 4-bit AND-circuit.

Figure 5 An $n$-bit tree-design AND-circuit.
**Solution.** Proof.** From the definition of depth-$n$ circuit, we have

$$\text{gate}(n + 1) = 2\text{gate}(n) + 1.$$  

for $n \geq 1$.

The induction hypothesis $P(n)$ is equation (4).

**Base case:** $(n = 1)$. A depth-1 circuit is just one AND-gate, so $\text{gate}(1) = 1 = 2^1 - 1$. So $P(1)$ is true.

**Inductive step:** For $n \geq 1$,

$$\text{gate}(n + 1) = 2\text{gate}(n) + 1 \quad \text{by definition of depth-$n$ circuit}$$

$$= 2(2^n - 1) + 1 \quad \text{by induction hypothesis}$$

$$= 2^{n+1} - 2 + 1 = 2^{n+1} - 1,$$

proving $P(n + 1)$ is true.

We conclude that (4) holds for all $n \geq 1$ as required.

---

**Problem 5.**

The Fibonacci numbers $F(0) F(1) F(2), \ldots$ are defined as follows:

$$F(n) := \begin{cases} 
0 & \text{if } n = 0, \\
1 & \text{if } n = 1, \\
F(n - 1) + F(n - 2) & \text{if } n > 1.
\end{cases}$$

Prove, using strong induction, the following closed-form formula for the Fibonacci numbers:

$$F(n) = \frac{p^n - q^n}{\sqrt{5}}$$

where $p = \frac{1+\sqrt{5}}{2}$ and $q = \frac{1-\sqrt{5}}{2}$.

**Hint:** Note that $p$ and $q$ are the roots of $x^2 - x - 1 = 0$, and so $p^2 = p + 1$ and $q^2 = q + 1$.

**COMMENTS:**

- CP_fibonacci_by_induction
- By Ali Kazerani, proof of Binet’s formula using strong induction
- edits & format by ARM 1/15/12

**keywords** = [ induction fibonacci recurrence closed_form Binet’s formula ]

**Solution.** Proof.** We will proceed by strong induction on $n$. Let the induction hypothesis $P(n)$ be that the given closed-form formula holds at $n$, that is,

$$F(n) = \frac{p^n - q^n}{\sqrt{5}}.$$
Base case \((n=0)\): \(P(0)\) is true, since
\[
\frac{p^n - q^n}{\sqrt{5}} = \frac{p^0 - q^0}{\sqrt{5}} = \frac{1 - 1}{\sqrt{5}} = 0 = F_0
\]

Base Case \((n=1)\): \(P(1)\) is true, since
\[
\frac{p^n - q^n}{\sqrt{5}} = \frac{p^1 - q^1}{\sqrt{5}} = \frac{p - q}{\sqrt{5}} = \frac{\sqrt{5}}{\sqrt{5}} = 1 = F_1.
\]

Inductive Step \((n>1)\): Since \(0 \leq n - 1, n < n + 1\), we may assume the strong induction hypothesis that \(P(n-1)\) and \(P(n)\) are both true. We will use this to prove \(P(n+1)\).

That is, we may assume
\[
F(n - 1) = \frac{p^{n-1} - q^{n-1}}{\sqrt{5}} \\
F(n) = \frac{p^n - q^n}{\sqrt{5}}
\]

From the hint we have that \(p^2 = p + 1\), which implies that \(p^2 p^{n-1} = (p + 1) p^{n-1}\) and so
\[
p^{n+1} = p^n + p^{n-1}.
\] (8)

Likewise \(q^2 = q + 1\), and so
\[
q^{n+1} = q^n + q^{n-1}
\] (9)

Subtracting (9) from (8) gives
\[
p^{n+1} - q^{n+1} = p^n - q^n + p^{n-1} - q^{n-1}
\]
and dividing by \(\sqrt{5}\) yields
\[
\frac{p^{n+1} - q^{n+1}}{\sqrt{5}} = \frac{p^n - q^n}{\sqrt{5}} + \frac{p^{n-1} - q^{n-1}}{\sqrt{5}}
\]
\[
= F_n + F_{n-1}
\] (by (5) and (6)) (10)

But \(F(n + 1) = F(n) + F(n - 1)\) for \(n > 1\) by definition, so (10) implies
\[
F(n + 1) = \frac{p^{n+1} - q^{n+1}}{\sqrt{5}}.
\]

That is, \(P(n + 1)\) is true in this case as well.

We conclude by strong induction that \(P(n)\) holds for all \(n \in \mathbb{N}\). ■

An alternative to Problem 5
(for those found the published solution to Problem 5)
Problem 6.
We examine a series of propositional formulas $F_1, F_2, \ldots, F_n, \ldots$ containing propositional variables $P_1, P_2, \ldots, P_n, \ldots$ constructed as follows

\[
\begin{align*}
F_1(P_1) & \::= \ P_1 \\
F_2(P_1, P_2) & \::= \ P_1 \ IMPLIES \ P_2 \\
F_3(P_1, P_2, P_3) & \::= \ (P_1 \ IMPLIES \ P_2) \ IMPLIES \ P_3 \\
F_4(P_1, P_2, P_3, P_4) & \::= \ ((P_1 \ IMPLIES \ P_2) \ IMPLIES \ P_3) \ IMPLIES \ P_4 \\
F_5(P_1, P_2, P_3, P_4, P_5) & \::= \ (((P_1 \ IMPLIES \ P_2) \ IMPLIES \ P_3) \ IMPLIES \ P_4) \ IMPLIES \ P_5 \\
& \vdots
\end{align*}
\]

Let $T_n$ be the number of different true/false settings of the variables $P_1, P_2, \ldots, P_n$ for which $F_n(P_1, P_2, \ldots, P_n)$ is true. For example, $T_2 = 3$ since $F_2(P_1, P_2)$ is true for 3 different settings of the variables $P_1$ and $P_2$:

<table>
<thead>
<tr>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$F_2(P_1, P_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$F$</td>
</tr>
<tr>
<td>$F$</td>
<td>$T$</td>
<td>$T$</td>
</tr>
<tr>
<td>$F$</td>
<td>$F$</td>
<td>$T$</td>
</tr>
</tbody>
</table>

(a) Explain why

\[ T_{n+1} = 2^{n+1} - T_n. \]  \hfill (11)

COMMENTS:

- FP_sat_count_induction
- same as FP_sat_count_genfunc but demands induction
- final.S05
- edited ARM 5/23/12, revised to justify given recurrence 3/9/17

keywords = [ induction satisfiability predicate proposition ]

Solution. We have:

\[ F_{n+1}(P_1, P_2, \ldots, P_{n+1}) = F_n(P_1, P_2, \ldots, P_n) \ IMPLIES \ P_{n+1} \]

If $P_{n+1}$ is true, then $F_{n+1}$ is true for all $2^n$ settings of the variables $P_1, P_2, \ldots, P_n$. If $P_{n+1}$ is false, then $F_{n+1}$ is true for all $2^n$ settings of $P_1, P_2, \ldots, P_n$ except for the $T_n$ settings that make $F_n$ true. Thus, altogether we have:

\[ T_{n+1} = 2^n + (2^n - T_n) = 2^{n+1} - T_n. \]

(b) Use induction to prove that

\[ T_n = \frac{2^{n+1} + (-1)^n}{3} \]  \hfill (*)

for $n \geq 1.$
Solution. Proof. The proof is by induction with $P(n)$ given by equation (*) as the induction hypothesis.

Base case: $(n = 1)$. There is a single setting of $P_1$ that makes $F_1(P_1) = P_1$ true, so

$$T_1 = 1 = \frac{2^{1+1} + (-1)^1}{3},$$

which proves $P(1)$.

Inductive step: For $n \geq 1$, we assume (*) and reason as follows:

$$T_{n+1} = 2^{n+1} - T_n$$

(by (11))

$$= 2^{n+1} - \frac{2^{n+1} + (-1)^n}{3}$$

(by ind. hyp.)

$$= \frac{2^{n+1} - 2^{n+1} - (-1)^n}{3}$$

$$= \frac{2^{n+2} + (-1)^{n+1}}{3},$$

so $P(n + 1)$ holds. \[\blacksquare\]