Concordia University

Department of Computer Science and Software Engineering

SOEN 331 - S and U

Introduction to Formal Methods for Software Engineering

Assignment 1 - Solutions

Propositional and Predicate Logic, Structures, Binary Relations, Functions and Relational Calculus

Team 19

Samuel Boaknin
40009692

Ryan Leyland
40015165

Saleha Tariq
40006997

Meng Susana Ung
40099729

Date: February 11, 2021

Contents

1 Solutions **3**

1.1 Problem 1 (8 pts) 3
1.2 Problem 2 (8 pts) 4
1.3 Problem 3 (12 pts) 5
1.4 Problem 4 (12 pts) 7
1.5 Problem 5 (12 pts) 8
1.6 Problem 6 (8 pts) 9
1.7 Problem 7 (8 pts) 10
1.8 Problem 8 (12 pts) 11
1.9 Problem 9 (20 pts) 12

1 Solutions

1.1 Problem 1 (8 pts)

You are shown a set of four cards placed on a table, each of which has a **number** on one side and a **symbol** on the other side. The visible faces of the cards show the numbers **2** and **7**, and the symbols $\square$, and $\bigcirc$.

Which card(s) must you turn over in order to test the truth of the proposition that *"If a card has an odd number on one side, then it has the symbol $\square$ on the other side"*? Explain your reasoning by deciding for <u>each</u> card whether it should be turned over and why.

Solution:

Cards to turn over: 7 and $\bigcirc$

Let p denote odd number on this side of card and q denote $\square$ on opposite side of card then:

7: check this card due to modus ponens

$p \to q, p \therefore q$

$\bigcirc$: check this card due to modus tollens

$p \to q, \neg q \therefore \neg p$

2: do not check this card due to inverse error

$p \to q, \neg p \therefore \neg q$ is incorrect

$\square$: do not check this card due to converse error

$p \to q, q \therefore p$ is also incorrect

$p \to q$ statements are disproven by finding $p \to !q$. Therefore, we should turn over cards who's visible face is p, which in this case is **7**.

The two cards with **2** facing up are not cards with **7**; therefore we should forgo turning those over. If we find a card with **7** facing up, and find a $\square$ on the other side upon flipping it, while it bodes well for the proposition, it does not prove it to be true. We must find that the other card with **7** facing up must **not** have a $\bigcirc$ on it.

Therefore, to prove the claim, one ought to flip both cards with **7** facing up, with the intent to find a counterexample.

1.2 Problem 2 (8 pts)

Consider the predicate *asks(a, b)* that is interpreted as *"a has asked b out on a date."*

1. Translate the following into English: $\forall a \, \exists b \;\; asks(a, b)$ and $\exists b \, \forall a \;\; asks(a, b)$.

2. Can we claim that $\forall a \, \exists b \;\; asks(a, b) \rightarrow \exists b \, \forall a \;\; asks(a, b)$? Discuss in detail.

3. Can we claim that $\exists b \, \forall a \;\; asks(a, b) \rightarrow \forall a \, \exists b \;\; asks(a, b)$? Discuss in detail.

Solution:

1. (a) $\forall a \, \exists b \;\; asks(a, b) =$ Everyone has asked at least one person out on a date.

   (b) $\exists b \, \forall a \;\; asks(a, b) =$ There exists a person whom has been asked on a date by everyone.

2. Can we claim that $\forall a \, \exists b \;\; asks(a, b) \rightarrow \exists b \, \forall a \;\; asks(a, b)$? Discuss in detail.

   No, modus ponens and modus tollens both fail. There could exist a person whom has not asked the same person out on a date as everyone else. Therefore the LHS would be true and the RHS false which cannot hold for implication.

3. Can we claim that $\exists b \, \forall a \;\; asks(a, b) \rightarrow \forall a \, \exists b \;\; asks(a, b)$? Discuss in detail.

   Yes, if a person has been asked on a date by every person then this implies that every person has asked out at least one person on a date.

1.3 Problem 3 (12 pts)

Let $scientist(x)$ denote the statement "x is a scientist", and $honest(x)$ denote the statement "x is honest." Formalize the following sentences and indicate their corresponding formal type.

1. "No scientists are honest."
2. "All scientists are crooked."
3. "All scientists are honest."
4. "Some scientists are crooked."
5. "Some scientists are honest."
6. "No scientist is crooked."
7. "Some scientists are not crooked."
8. "Some scientists are not honest."

Identify pairs that are contradictories, contraries, subcontraries, and pairs that support subalteration (clearly indicating superaltern and subaltern).

Solution:

The formalization is as follows:

1. $\forall x(\ scientist(x) \Rightarrow \neg \ honest(x))$ (Type E)
2. $\forall x(\ scientist(x) \Rightarrow \neg \ honest(x))$ (Type E)
3. $\forall x(\ scientist(x) \Rightarrow \ honest(x))$ (Type A)
4. $\exists x(\ scientist(x) \wedge \neg \ honest(x))$ (Type O)
5. $\exists x(\ scientist(x) \wedge \ honest(x))(TypeI)$
6. $\forall x(\ scientist(x) \Rightarrow \ honest(x))$ (Type A)

7. $\exists x(\ \mathit{scientist}(x) \wedge \ \mathit{honest}(x))$ (Type I)

8. $\exists x(\ \mathit{scientist}(x) \wedge \neg\, \mathit{honest}(x))$ (Type O)

Statements 1 & 2 are contradictory with Statements 5 & 7. Statements 3 & 6 are contradictory with Statements 4 & 8.

Statements 1 & 2 are contrary with Statements 3 & 6. Statements 4 & 8 are subcontrary with Statements 5 & 7.

Statements 1 & 2 are superalterns to the subaltern Statements 4 & 8. Statements 3 & 6 are superalterns to the subaltern Statements 5 & 7.

1.4 Problem 4 (12 pts)

Consider list $\Lambda = \langle w, x, y, z\rangle$, deployed to implement a Queue Abstract Data Type.

1. Let the head of Λ correspond to the front position of the Queue. Implement operations `enqueue(el, Λ)` and `dequeue(Λ)` using list construction operations. In both cases we can refer to Λ' as the state of the list upon successful termination of one of its operations.

2. Let us now reverse the way we manipulate our data structure and let the head of Λ correspond to the rear of the Queue.

   (a) What would be the result of $cons(el, \Lambda)$, and would it be a correct implementation for operation `enqueue(el, Λ)`?

   (b) What would be the result of $list(el, \Lambda)$, and would it be a correct implementation for operation `enqueue(el, Λ)`?

   (c) What would be the result of $concat(list(el), \Lambda)$, and would it be a correct implementation for operation `enqueue(el, Λ)`?

Solution:

1. (a) `enqueue(el, Λ)` : Λ' = (append Λ (list el))

   (b) `dequeue(Λ)` : let temp = (car Λ); Λ' = (cdr Λ); return temp;

2. (a) $\Lambda' = \langle el, w, x, y, z\rangle$, Yes correct as new element added to tail of queue

   (b) $\Lambda' = \langle el, \langle w, x, y, z\rangle\rangle$, No 'list' function takes elements as parameters, so now there are only two direct elements inside Λ'

   (c) $\Lambda' = \langle el, w, x, y, z\rangle$, Yes correct order, new element added to tail of queue

1.5 Problem 5 (12 pts)

Let $A = \{0, 1, 2, 3, 4\}$ and relations R, S, T, and U on A defined as follows:

$$R = \{(0,0), (0,1), (0,3), (1,0), (1,1), (2,2), (3,0), (3,1), (3,3), (4,0), (4,1), (4,3), (4,4)\}$$

$$S = \{(0,1), (1,1), (2,3), (2,4), (3,0), (3,4), (4,0), (4,1), (4,4)\}$$

$$T = \{(0,3), (0,4), (2,1), (3,2), (4,2), (4,3)\}$$

$$U = \{(0,0), (0,1), (0,3), (1,0), (1,1), (1,3), (2,2), (3,0), (3,1), (3,3), (4,4)\}$$

Fill in the table below, using ✓, or ×.

		R	S	T	U
Reflexive		✓	×	×	✓
Irreflexive		×	×	✓	×
Symmetric		×	×	×	✓
Asymmetric		×	×	✓	×
Antisymmetric		×	✓	✓	×
Transitive		×	×	×	✓
Equivalence		×	×	×	✓
Partial order		×	×	×	×

1.6 Problem 6 (8 pts)

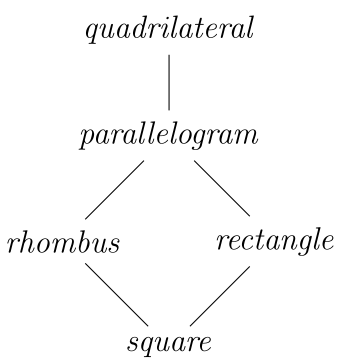
Consider the relation *"is a subtype of"* over the set

$$\{rectangle, quadrilateral, square, parellelogram, rhombus\}.$$

1. Is this an *equivalence relation*?
2. Is this relation a *partial order*? If so, create a *Hasse diagram*, and identify *minimal* and *maximal* elements.

Solution:

1. No not an equivalence relation. An equivalence relation must be reflexive, symmetric and transitive. The relation is not symmetric, for example a square is a subtype of a quadrilateral but a quadrilateral is not a subtype of a square as a square is defined to have four right angled corners.
2. Yes, maximal = quadrilateral & minimal = square



1.7 Problem 7 (8 pts)

Consider the set $A = \{w, x, y, z\}$, and the relations

$S = \{(w, x), (w, y), (x, w), (x, x), (z, x)\}$

$T = \{(w, w), (w, y), (x, w), (x, x), (x, z), (y, w), (y, y), (y, z)\}$

Find the following compositions:

1. $S \circ T$

   $S \circ T = \{(w, w), (w, x), (w, y), (w, z), (x, w), (x, x), (x, y), (x, z), (z, w), (z, x), (z, z)\}$

2. $T \circ S$

   $T \circ S = \{(w, x), (w, y), (x, w), (x, x), (x, y), (y, x), (y, y)\}$

3. $T^{-1} \circ S^{-1}$

   $T^{-1} \circ S^{-1} = \{(w, w), (w, x), (w, z), (x, w), (x, x), (x, z), (y, w), (y, x), (z, w), (z, x), (z, z)\}$

NOTE: Some authors (e.g. Rosen) adopt a different ordering of operands than the one we use in our lecture notes. Please follow the ordering (and the definition) of the lecture notes.

1.8 Problem 8 (12 pts)

Consider sets $A = \{1, 2, 3, 4, 5, 6\}$ and $B = \{a, b, c, d, e, f\}$.

1. Determine the type of the correspondence in each of the following cases, or indicate if the correspondence is not a function.

   (a) $\{1 \mapsto b, 2 \mapsto c, 3 \mapsto e, 4 \mapsto d, 5 \mapsto f, 3 \mapsto a\}$

   (b) $\{1 \mapsto a, 2 \mapsto d, 3 \mapsto a, 4 \mapsto f, 5 \mapsto d, 6 \mapsto c\}$

   (c) $\{1 \mapsto c, 2 \mapsto b, 3 \mapsto d, 4 \mapsto e, 5 \mapsto e, 6 \mapsto f\}$

   (d) $\{1 \mapsto b, 2 \mapsto c, 3 \mapsto e, 4 \mapsto d, 5 \mapsto f, 6 \mapsto a\}$

   Fill in the table below, using ✓, or ×.

	Injective	Surjective	Bijective	Neither injective nor surjective	Not a function
(a)	×	×	×	×	✓
(b)	×	×	×	✓	×
(c)	×	×	×	✓	×
(d)	✓	✓	✓	×	×

2. Is it possible to construct a function $f : A \to B$ which is surjective and not injective? Discuss.

No, it is not possible since there are 6 elements in the domain and 6 elements in the codomain. In order for the function to be surjective, each element of the codomain has to be mapped to by at least one element of the domain. If the function were to not be injective, then distinct elements of the domain would be mapped to the same element of the codomain which means that some element or elements of the codomain would not be mapped to by at least one element of the domain since there are equal amounts of elements in the domain and codomain, thus it isn't possible for the surjective function to not be injective.

1.9 Problem 9 (20 pts)

Consider the following relation:

$$laptops : Model \leftrightarrow Brand$$

where

$$
\begin{array}{l}
laptops = \\
\quad \{ \\
\quad\quad legion5 \mapsto lenovo, \\
\quad\quad macbookair \mapsto apple, \\
\quad\quad xps15 \mapsto dell, \\
\quad\quad spectre \mapsto hp, \\
\quad\quad xps13 \mapsto dell, \\
\quad\quad swift3 \mapsto acer, \\
\quad\quad macbookpro \mapsto apple, \\
\quad\quad dragonfly \mapsto hp, \\
\quad\quad envyx360 \mapsto hp \\
\quad \}
\end{array}
$$

1. What is the domain and the range of the relation?

   $$\operatorname{dom} laptops = \{legion5, macbookair, xps15, spectre, xps13, swift3, macbookpro, dragonfly, envyx360\}$$

   $$\operatorname{ran} laptops = \{lenovo, apple, dell, hp, acer\}$$

2. What is the result of the expression

   $$\{xps15, xps13, swift3, envyx360\} \lhd laptops$$

   $$
   \begin{array}{l}
   \{xps15, xps13, swift3, envyx360\} \lhd laptops = \\
   \quad \{ \\
   \quad\quad xps15 \mapsto dell, \\
   \quad\quad xps13 \mapsto dell, \\
   \quad\quad swift3 \mapsto acer, \\
   \quad\quad envyx360 \mapsto hp \\
   \quad \}
   \end{array}
   $$

What is the meaning of operator $\lhd$ and where would you deploy such operator in the context of a database management system?

The $\lhd$ operator restricts the domain of the relation and returns the pairs from the relation where the first element matches one of the elements in the domain restrictor. In the context of a database management system, this domain restriction operator could be used to query the database to determine which *Brands* correspond to certain *Models*, given a specific set of *Models*.

3. What is the result of the expression

$$laptops \rhd \{lenovo, hp\}$$

$$\begin{array}{l} laptops \rhd \{lenovo, hp\} = \\ \quad \{ \\ \qquad legion5 \mapsto lenovo, \\ \qquad spectre \mapsto hp, \\ \qquad dragonfly \mapsto hp, \\ \qquad envyx360 \mapsto hp \\ \quad \} \end{array}$$

What is the meaning of operator $\rhd$ and where would you deploy such operator in the context of a database management system?

The $\rhd$ operator restricts the range of the relation and returns the pairs from the relation where the second element (*Brand*) matches one of the elements in the range restrictor. In the context of a database management system, this range restriction operator could be used to query the database to determine which *Models* correspond to certain *Brands*, given a specific set of *Brands*.

4. What is the result of the expression

$$\{legion5, xps15, xps13, dragonfly\} ⩤ laptops$$

$$
\begin{aligned}
&\{legion5, xps15, xps13, dragonfly\} \lhd laptops = \\
&\quad \{ \\
&\qquad macbookair \mapsto apple, \\
&\qquad spectre \mapsto hp, \\
&\qquad swift3 \mapsto acer, \\
&\qquad macbookpro \mapsto apple, \\
&\qquad envyx360 \mapsto hp \\
&\quad \}
\end{aligned}
$$

What is the meaning of operator $\lhd$ and where would you deploy such operator in the context of a database management system?

The $\lhd$ operator performs a domain subtration and removes all pairs from the domain of the relation based on their first elements. In the context of a database management system, this operator could be used to remove pairs from the database based on the name of the first element. Here, we removed *laptops* from the domain of the relation whose *Model* matched the ones with the operator.

5. What is the result of the expression

$$laptops \rhd \{apple, dell, hp\}$$

$$
\begin{aligned}
&laptops \rhd \{apple, dell, hp\} = \\
&\quad \{ \\
&\qquad legion5 \mapsto lenovo, \\
&\qquad swift3 \mapsto acer, \\
&\quad \}
\end{aligned}
$$

What is the meaning of operator $\rhd$ and where would you deploy such operator in the context of a database management system?

The $\rhd$ operator performs a range subtration and removes all pairs from the codomain of the relation based on their second elements. In the context of a database management system, this operator could be used to remove pairs from the database based on the name of the second element. Here, we removed *laptops* from the codomain of the relation whose *Brand* matched the ones with the operator.

6. Consider the following expression

$$laptops \oplus \{ideapad \mapsto lenovo\}$$

(a) What is the result of the expression?

$$
\begin{array}{l}
laptops \oplus \{ideapad \mapsto lenovo\} = \\
\quad \{ \\
\qquad legion5 \mapsto lenovo, \\
\qquad macbookair \mapsto apple, \\
\qquad xps15 \mapsto dell, \\
\qquad spectre \mapsto hp, \\
\qquad xps13 \mapsto dell, \\
\qquad swift3 \mapsto acer, \\
\qquad macbookpro \mapsto apple, \\
\qquad dragonfly \mapsto hp, \\
\qquad envyx360 \mapsto hp \\
\qquad ideapad \mapsto lenovo \\
\quad \}
\end{array}
$$

The relation will have $ideapad \mapsto lenovo$ appended to it.

(b) What is the meaning of operator $\oplus$ and where would you deploy such operator in the context of a database management system?

The $\oplus$ operator performs a relational override that adds all elements from the RHS of the operator to the relation, if they do not already exist in the relation. In the context of a database management system, this could be used to update the database and add new elements.

(c) Does the result of the expression have a permanent effect on the database (relation)? If not, describe in detail how would you ensure a permanent effect.

This result would not have a permanent effect on the relation, it will only return what the relation would look like with the elements added to it. For a permanent effect, you would need to do

$$laptops' = laptops \oplus \{ideapad \mapsto lenovo\}$$

END OF ASSIGNMENT SOLUTIONS.