COMPSCI/SFWRENG 2FA3

Discrete Mathematics with Applications II

Winter 2020

Week 05 Exercises with Solutions

Dr. William M. Farmer

McMaster University

Revised: Feb 9, 2020

Background Definitions

Consider the following definitions:

1. $\Sigma_{\text{mon}} = (\{M\}, \{e\}, \{*\}, \emptyset, \tau)$ where $\tau(e) = M$ and $\tau(*) = M \times M \to M$.

2. Let Γ_{mon} be the following set of Σ-sentences:

$$\begin{array}{rl} \text{Assoc} & \forall\, x, y, z : M \,.\, (x * y) * z = x * (y * z). \\ \text{IdLeft} & \forall\, x : M \,.\, e * x = x. \\ \text{IdRight} & \forall\, x : M \,.\, x * e = x. \end{array}$$

3. $T_{\text{mon}} = (\Sigma_{\text{mon}}, \Gamma_{\text{mon}})$.

4. $\mathcal{M}_{\text{nat}}$ is the Σ_{mon}-structure derived from $(\mathbb{N}, 0, +)$.

5. $\mathcal{M}_{\text{triv}}$ is the Σ_{mon}-structure derived from the *trivial monoid* $(\{0\}, 0, +)$.

6. $\Sigma_{\text{grp}} = (\{G\}, \{e\}, \{*, \mathsf{inv}\}, \emptyset, \tau)$ where $\tau(e) = G$, $\tau(*) = G \times G \to G$, and $\tau(\mathsf{inv}) = G \to G$.

7. Let Γ_{grp} be the following set of Σ-sentences:

$$\begin{array}{rl} \text{Assoc} & \forall\, x, y, z : G \,.\, (x * y) * z = x * (y * z). \\ \text{IdLeft} & \forall\, x : G \,.\, e * x = x. \\ \text{IdRight} & \forall\, x : G \,.\, x * e = x. \\ \text{InvLeft} & \forall\, x : G \,.\, \mathsf{inv}(x) * x = e. \\ \text{InvRight} & \forall\, x : G \,.\, x * \mathsf{inv}(x) = e. \end{array}$$

8. $T_{\text{grp}} = (\Sigma_{\text{grp}}, \Gamma_{\text{grp}})$.

9. $\Sigma_{\text{stack}} = (\mathcal{B}, \mathcal{C}, \mathcal{F}, \mathcal{P}, \tau)$ where:

   a. $\mathcal{B} = \{\mathsf{Element}, \mathsf{Stack}\}$.
   b. $\mathcal{C} = \{\mathsf{error}, \mathsf{bottom}\}$.
   c. $\mathcal{F} = \{\mathsf{push}, \mathsf{pop}, \mathsf{top}\}$.
   d. $\mathcal{P} = \emptyset$.
   e. $\tau(\mathsf{error}) = \mathsf{Element}$.
   f. $\tau(\mathsf{bottom}) = \mathsf{Stack}$.
   g. $\tau(\mathsf{push}) = \mathsf{Element} \times \mathsf{Stack} \to \mathsf{Stack}$.
   h. $\tau(\mathsf{pop}) = \mathsf{Stack} \to \mathsf{Stack}$.
   i. $\tau(\mathsf{top}) = \mathsf{Stack} \to \mathsf{Element}$.

Exercises

1. Let $\Sigma = (\alpha, a : \alpha, f : \alpha \times \alpha \to \alpha, p : \alpha \times \alpha \to \mathbb{B})$. Compute fvar and bvar for each of the following Σ-formulas:

   a. $\exists\, x : \alpha \,.\, \exists\, y : \alpha \,.\, p(z : \alpha)$.
   SOLUTION:
   Easy way: We can simply observe that $z : \alpha$ is not bounded by $\forall$ or $\exists$ and that $x : \alpha$ and $y : \alpha$ are. Thus fvar is $\{z : \alpha\}$ and bvar is $\{x : \alpha, y : \alpha\}$.

   Long way: We'll use the definitions of fvar and bvar given via pattern matching.
   First we'll find fvar and bvar for basic formulas using existential quantifier, logical and, and logical or (we'll use these for part a and b as well).

$$
\begin{aligned}
& \mathsf{fvar}(\exists x : \alpha. A) & \qquad & \mathsf{bvar}(\exists x : \alpha. A) \\
= \;& \mathsf{fvar}(\neg(\forall x : \alpha. \neg A)) & = \;& \mathsf{bvar}(\neg(\forall x : \alpha. \neg A)) \\
= \;& \mathsf{fvar}(\forall x : \alpha. \neg A) & = \;& \mathsf{bvar}(\forall x : \alpha. \neg A) \\
= \;& \mathsf{fvar}(\neg A) \backslash \{x : \alpha\} & = \;& \mathsf{bvar}(\neg A) \cup \{x : \alpha\} \\
= \;& \mathsf{fvar}(A) \backslash \{x : \alpha\} & = \;& \mathsf{bvar}(A) \cup \{x : \alpha\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{fvar}(A \vee B) & \qquad & \mathsf{bvar}(A \vee B) \\
= \;& \mathsf{fvar}(\neg A \Rightarrow B) & = \;& \mathsf{bvar}(\neg A \Rightarrow B) \\
= \;& \mathsf{fvar}(\neg A) \cup \mathsf{fvar}(B) & = \;& \mathsf{bvar}(\neg A) \cup \mathsf{bvar}(B) \\
= \;& \mathsf{fvar}(A) \cup \mathsf{fvar}(B) & = \;& \mathsf{bvar}(A) \cup \mathsf{bvar}(B)
\end{aligned}
$$

$$\begin{aligned}
&\mathsf{fvar}(A \wedge B) \\
= \;& \mathsf{fvar}(\neg(\neg A \vee \neg B)) \\
= \;& \mathsf{fvar}(\neg A \vee \neg B) \\
= \;& \mathsf{fvar}(\neg A) \cup \mathsf{fvar}(\neg B) \\
= \;& \mathsf{fvar}(A) \cup \mathsf{fvar}(B)
\end{aligned}
\qquad
\begin{aligned}
&\mathsf{bvar}(A \wedge B) \\
= \;& \mathsf{bvar}(\neg(\neg A \vee \neg B)) \\
= \;& \mathsf{bvar}(\neg A \vee \neg B) \\
= \;& \mathsf{bvar}(\neg A) \cup \mathsf{bvar}(\neg B) \\
= \;& \mathsf{bvar}(A) \cup \mathsf{bvar}(B)
\end{aligned}$$

Now we will find fvar and bvar:

$$\begin{aligned}
&\mathsf{fvar}(\exists\, x : \alpha \;.\; \exists\, y : \alpha \;.\; p(z : \alpha)) \\
= \;& \mathsf{fvar}(\exists\, y : \alpha \;.\; p(z : \alpha)) \backslash \{x : \alpha\} \\
= \;& (\mathsf{fvar}(p(z : \alpha)) \backslash \{y : \alpha\}) \backslash \{x : \alpha\} \\
= \;& (\mathsf{fvar}(z : \alpha) \backslash \{y : \alpha\}) \backslash \{x : \alpha\} \\
= \;& (\{z : \alpha\} \backslash \{y : \alpha\}) \backslash \{x : \alpha\} \\
= \;& \{z : \alpha\}
\end{aligned}$$

$$\begin{aligned}
&\mathsf{bvar}(\exists\, x : \alpha \;.\; \exists\, y : \alpha \;.\; p(z : \alpha)) \\
= \;& \mathsf{bvar}(\exists\, y : \alpha \;.\; p(z : \alpha)) \cup \{x : \alpha\} \\
= \;& (\mathsf{bvar}(p(z : \alpha)) \cup \{y : \alpha\}) \cup \{x : \alpha\} \\
= \;& (\emptyset \cup \{y : \alpha\}) \cup \{x : \alpha\} \\
= \;& \{y : \alpha, x : \alpha\}
\end{aligned}$$

b. $f(x : \alpha) = a \wedge$
$\forall\, y : \alpha \;.\; ((p(y : \alpha) \vee p(x : \alpha)) \Rightarrow \exists\, x : \alpha \;.\; p(f(x : \alpha)))$.
Easy way: We can simply observe that $x : \alpha$ is not bounded by $\forall$ or $\exists$ in its first occurrence and $x : \alpha$ and $y : \alpha$ both have an occurrence bounded by $\forall$ or $\exists$ after the conjunction. Thus fvar is $\{x : \alpha\}$ and bvar is $\{x : \alpha, y : \alpha\}$.

Long way: We'll use the definitions of fvar and bvar given via pattern matching.

For clarity, let $A \equiv \forall\, y : \alpha \;.\; ((p(y : \alpha) \vee p(x : \alpha)) \Rightarrow \exists\, x : \alpha \;.$

$p(f(x : \alpha)))$

$$
\begin{aligned}
& \mathsf{fvar}(f(x : \alpha) = a \wedge A) \\
= \;& \mathsf{fvar}(f(x : \alpha) = a) \cup \mathsf{fvar}(A) \\
= \;& \mathsf{fvar}(f(x : \alpha)) \cup \mathsf{fvar}(a) \cup \mathsf{fvar}(A) \\
= \;& \mathsf{fvar}(x : \alpha) \cup \mathsf{fvar}(a) \cup \mathsf{fvar}(A) \\
= \;& \{x : \alpha\} \cup \mathsf{fvar}(a) \cup \mathsf{fvar}(A) \\
= \;& \{x : \alpha\} \cup \emptyset \cup \mathsf{fvar}(A) \\
= \;& \{x : \alpha\} \cup \mathsf{fvar}(A) \\
= \;& \{x : \alpha\} \cup \\
& \quad (\mathsf{fvar}((p(y : \alpha) \vee p(x : \alpha)) \Rightarrow \exists\, x : \alpha \,.\, p(f(x : \alpha))) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\mathsf{fvar}((p(y : \alpha) \vee p(x : \alpha))) \cup \mathsf{fvar}(\exists\, x : \alpha \,.\, p(f(x : \alpha)))) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\mathsf{fvar}(p(y : \alpha)) \cup \mathsf{fvar}(p(x : \alpha)) \cup \mathsf{fvar}(\exists\, x : \alpha \,.\, p(f(x : \alpha)))) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\mathsf{fvar}(y : \alpha) \cup \mathsf{fvar}(x : \alpha) \cup \mathsf{fvar}(\exists\, x : \alpha \,.\, p(f(x : \alpha)))) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup \mathsf{fvar}(\exists\, x : \alpha \,.\, p(f(x : \alpha)))) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup (\mathsf{fvar}(p(f(x : \alpha))) \backslash \{x : \alpha\})) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup (\mathsf{fvar}(f(x : \alpha)) \backslash \{x : \alpha\})) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup (\mathsf{fvar}(x : \alpha) \backslash \{x : \alpha\})) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup (\{x : \alpha\} \backslash \{x : \alpha\})) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \\
& \quad ((\{y : \alpha\} \cup \{x : \alpha\} \cup \emptyset) \backslash \{y : \alpha\}) \\
= \;& \{x : \alpha\} \cup \{x : \alpha\} \\
= \;& \{x : \alpha\}
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{bvar}(f(x:\alpha) = a \wedge A) \\
= \;& \mathsf{bvar}(f(x:\alpha) = a) \cup \mathsf{bvar}(A) \\
= \;& \emptyset \cup \mathsf{bvar}(A) \\
= \;& \mathsf{bvar}(\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha)))) \\
= \;& \mathsf{bvar}((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \mathsf{bvar}(p(y:\alpha) \vee p(x:\alpha)) \cup \mathsf{bvar}(\exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \mathsf{bvar}(p(y:\alpha) \vee p(x:\alpha)) \cup \mathsf{bvar}(\exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \mathsf{bvar}(p(y:\alpha)) \cup \mathsf{bvar}(p(x:\alpha)) \cup \mathsf{bvar}(\exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \emptyset \cup \emptyset \cup \mathsf{bvar}(\exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \mathsf{bvar}(\exists\, x:\alpha\,.\,p(f(x:\alpha))) \cup \{y:\alpha\} \\
= \;& \mathsf{bvar}(p(f(x:\alpha))) \cup \{x:\alpha\} \cup \{y:\alpha\} \\
= \;& \emptyset \cup \{x:\alpha\} \cup \{y:\alpha\} \\
= \;& \{x:\alpha, y:\alpha\}
\end{aligned}
$$

2. Compute the following substitutions:

   a. $f(x:\alpha) = a \wedge$
   $\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha)))[x \mapsto f(a)]$.
   SOLUTION: (Note: the first free x is not included in the substitution; it's only on the body of the $\forall$)

   $$f(x:\alpha) = a \wedge (\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(f(a))) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha))))$$

   b. $f(x:\alpha) = a \wedge$
   $\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha)))[y \mapsto f(a)]$.
   SOLUTION: We first note that the occurrences of $y:\alpha$ is bound in the above expression where the substitution were to occur. So, we compute the substitution and get:

   $$f(x:\alpha) = a \wedge (\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha))))$$

   after substitution (i.e. there was no change).

   c. $f(x:\alpha) = a \wedge$
   $\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha)))[z \mapsto f(a)]$.
   SOLUTION: We first note that there are no occurrences of z. So, we compute the substitution and get:

   $$f(x:\alpha) = a \wedge (\forall\, y:\alpha\,.\,((p(y:\alpha) \vee p(x:\alpha)) \Rightarrow \exists\, x:\alpha\,.\,p(f(x:\alpha))))$$

   after substitution (i.e. there was no change).

3. Construct a signature of MSFOL that is suitable for formalizing:

   a. A queue of abstract elements.
      Solution:
      Let $\Sigma_{\mathsf{queue}} = (\mathcal{B}, \mathcal{C}, \mathcal{F}, \mathcal{P}, \tau)$ where:

      i. $\mathcal{B} = \{\mathsf{Element}, \mathsf{Queue}\}$.
      ii. $\mathcal{C} = \{\mathsf{error}, \mathsf{empty}\}$.
      iii. $\mathcal{F} = \{\mathsf{front}, \mathsf{back}, \mathsf{enqueue}, \mathsf{dequeue}\}$.
      iv. $\mathcal{P} = \emptyset$.
      v. $\tau(\mathsf{error}) = \mathsf{Element}$.
      vi. $\tau(\mathsf{empty}) = \mathsf{Queue}$.
      vii. $\tau(\mathsf{front}) = \mathsf{Queue} \rightarrow \mathsf{Element}$.
      viii. $\tau(\mathsf{back}) = \mathsf{Queue} \rightarrow \mathsf{Element}$.
      ix. $\tau(\mathsf{enqueue}) = \mathsf{Element} \times \mathsf{Queue} \rightarrow \mathsf{Queue}$.
      x. $\tau(\mathsf{dequeue}) = \mathsf{Queue} \rightarrow \mathsf{Queue}$.

   b. An abstract field.
      Solution:
      Let $\Sigma_{\mathsf{absField}} = (\mathcal{B}, \mathcal{C}, \mathcal{F}, \mathcal{P}, \tau)$ where:

      i. $\mathcal{B} = \{\mathsf{F}\}$.
      ii. $\mathcal{C} = \{0_{\mathsf{F}}, 1_{\mathsf{F}}\}$.
      iii. $\mathcal{F} = \{+_{\mathsf{F}}, *_{\mathsf{F}}, -_{\mathsf{F}}, \ _{\mathsf{F}}^{-1}\}$.
      iv. $\mathcal{P} = \emptyset$.
      v. $\tau(0_{\mathsf{F}}) = \tau(1_{\mathsf{F}}) = \mathsf{F}$.
      vi. $\tau(-_{\mathsf{F}}) = \tau(\ _{\mathsf{F}}^{-1}) = \mathsf{F} \rightarrow \mathsf{F}$.
      vii. $\tau(*_{\mathsf{F}}) = \tau(+_{\mathsf{F}}) = \mathsf{F} \times \mathsf{F} \rightarrow \mathsf{F}$.

   c. An abstract vector space over an abstract field.
      Solution:
      Let $\Sigma_{\mathsf{vecSpace}} = (\mathcal{B}, \mathcal{C}, \mathcal{F}, \mathcal{P}, \tau)$ where:

      i. $\mathcal{B} = \{\mathsf{F}, \mathsf{V}\}$.
      ii. $\mathcal{C} = \{0_{\mathsf{F}}, 1_{\mathsf{F}}, 0_{\mathsf{V}}\}$.
      iii. $\mathcal{F} = \{+_{\mathsf{F}}, *_{\mathsf{F}}, -_{\mathsf{F}}, \ _{\mathsf{F}}^{-1}, +_{\mathsf{V}}, -_{\mathsf{V}}, *_{\mathsf{V}}\}$.
      iv. $\mathcal{P} = \emptyset$.
      v. $\tau(0_{\mathsf{F}}) = \tau(1_{\mathsf{F}}) = \mathsf{F}$.
      vi. $\tau(-_{\mathsf{F}}) = \tau(\ _{\mathsf{F}}^{-1}) = \mathsf{F} \rightarrow \mathsf{F}$.
      vii. $\tau(*_{\mathsf{F}}) = \tau(+_{\mathsf{F}}) = \mathsf{F} \times \mathsf{F} \rightarrow \mathsf{F}$.
      viii. $\tau(0_{\mathsf{V}}) = \mathsf{V}$.
      ix. $\tau(-_{\mathsf{V}}) = \mathsf{V} \rightarrow \mathsf{V}$.
      x. $\tau(+_{\mathsf{V}}) = \mathsf{V} \times \mathsf{V} \rightarrow \mathsf{V}$.
      xi. $\tau(*_{\mathsf{V}}) = \mathsf{F} \times \mathsf{V} \rightarrow \mathsf{V}$.

4. Let $\Sigma_{\text{ord}} = (\mathcal{B}, \mathcal{C}, \mathcal{F}, \mathcal{P}, \tau)$ be the signature defined in the lecture slides. Construct Σ_{ord}-structures that define the following mathematical structures: $(\mathbb{N}, \leq)$, $(\mathbb{Z}, <)$, $(\mathbb{Q}, >)$, and $(\mathbb{R}, \neq)$.

   Solution:

   $\Sigma_{\text{ord}} = (\{U\}, \emptyset, \emptyset, \{<\}, \tau)$ where $\tau(<) = U \times U \to \mathbb{B}$. Let $\mathcal{M}_{\mathbb{N}} = (\{D_U\}, I)$ where $D_U = \mathbb{N}$ and $I(<)(m, n) = m \leq n$ for all $m, n \in \mathbb{N}$. $\mathcal{M}_{\mathbb{N}}$ is a Σ_{ord}-structure defining $(\mathbb{N}, \leq)$.

   The other Σ_{ord}-structures are constructed in a similar manner.

5. Construct a Σ_{stack}-structure such that $D_{\mathsf{Element}} = \mathbb{N}$, D_{Stack} is the set of finite sequences of members of $\mathbb{N}$, and the function symbols of Σ_{stack} manipulate the members of D_{Stack} as stacks.
   Solution:
   An Σ-structure is a pair $\mathcal{M} = (D, I)$

   Let the following be the signature of a language for stacks.

   $\Sigma_{\text{stack}} = (\{\text{Element}, \text{Stack}\}, \{\text{error}, \text{bottom}\}, \{push, pop, top\}, \emptyset, \tau)$

   where
   $\tau(\text{error}) = \text{Element}$ and $\tau(\text{bottom}) = \text{Stack}$ and
   $\tau(\text{push}) = \text{Element} \times \text{Stack} \to \text{Stack}$.
   $\tau(\text{pop}) = \text{Stack} \to \text{Stack}$
   $\tau(\text{top}) = \text{Stack} \to \text{Element}$

   Let $\mathcal{M}_{\text{stack}}$ be the Σ_{stack}-structure derived from the following:

   $\mathcal{M}_{\text{stack}} = (\mathbb{N}, \{\mathbb{N}\}, \text{error}, \text{bottom}, \text{push}, \text{pop}, \text{top})$.

   Where D is a collection $\{D_{\mathsf{Stack}}, D_{\mathsf{Element}} \| \mathsf{Stack}, \mathsf{Element} \in \mathcal{B}\}$

   and

   $D_{\mathsf{Stack}} = \{\mathbb{N}\}$
   $D_{\mathsf{Element}} = \mathbb{N}$

   and

   $I(\text{error}) \in \text{D}_{\mathsf{Element}}$
   $I(\text{bottom}) \in \text{D}_{\mathsf{Stack}}$
   $I(\text{push}) : \text{D}_{\mathsf{Element}} \times \text{D}_{\mathsf{Stack}} \to \text{D}_{\mathsf{Stack}}$
   $I(\text{pop}) : \text{D}_{\mathsf{Stack}} \to \text{D}_{\mathsf{Stack}}$
   $I(\text{top}) : \text{D}_{\mathsf{Stack}} \to \text{D}_{\mathsf{Element}}$

Therefore, with the structure defined above:

$I(\text{error}) = \text{error}$
$I(\text{bottom}) = \text{bottom}$
$I(\text{push}) = \text{push}$
$I(\text{pop}) = \text{pop}$
$I(\text{top}) = \text{top}$

6. Which of the following Σ_{mon}-formulas are satisfiable and which are universally valid?

   a. $e = e$.
   Solution:
   Universally valid. "=" is reflexive, regardless of the interpretation.

   b. $e = e * e$.
   Solution:
   Satisfiable. Both $\mathcal{M}_{\text{nat}}$ and $\mathcal{M}_{\text{triv}}$ satisfy $e = e * e$, and valid in both because, $0 = 0 + 0$. Not universally valid since the structure $(\mathbb{N}, 1, +)$ does not satisfy the formula since $1 = 1 + 1 \equiv \texttt{False}$

   c. $\forall x : M \, . \, x = e$.
   Solution:
   Satisfiable. It is satisfied by and valid for $\mathcal{M}_{\text{triv}}$ since

   $$(\forall x : \{0\} \, . \, x = 0) \equiv \texttt{True}$$

   $\mathcal{M}_{\text{nat}}$ does not satisfy the formula since

   $$(\forall x : \mathbb{N} \, . \, x = 0) \equiv \texttt{False}$$

   d. $\forall x : M \, . \, x \neq e$.
   Solution:
   Neither satisfiable nor universally valid (follows from the universal validity of a.)

7. Which of the following Σ_{mon}-formulas are valid in $\mathcal{M}_{\text{nat}}$ and which are valid in $\mathcal{M}_{\text{triv}}$?

   a. $e = e$.
   Solution:
   Valid.

   b. $e = e * e$.
   Solution:
   Valid. Follows from IdLeft and IdRight.

c. $\forall x : M . x = e$.
Solution:
Valid in $\mathcal{M}_{\text{triv}}$. Not valid in $\mathcal{M}_{\text{nat}}$.

d. $\forall x : M . x \neq e$.
Solution:
Not valid.

8. Which are the following Σ_{mon}-formulas are valid in T_{mon}.

   a. $e = e$.
   Solution:
   Valid (Universally as given in 1a.)

   b. $e = e * e$.
   Solution:
   Valid. Follows from IdLeft and $\mathsf{IdRight}$

   c. $\forall x : M . x = e$.
   Solution:
   Not Valid. $\mathcal{M}_{\text{nat}}$ is a model of T_{mon}, and this is not valid in $\mathcal{M}_{\text{nat}}$.

   d. $\forall x : M . x \neq e$.
   Solution:
   Not Valid. (Follows from 1d since it isn't satisfiable).