Homework 9: Final comprehensive homework

Changelog: This is version 3, updated June 8 9:00 AM. Fixed problem 6 example ($7 \cdot 5$ is 35).
Version 2: Number of points in problem 2 updated/clarified.

Due date: Wednesday June 9 at 11:00 PM (Seattle time, i.e. GMT-7)
If you work with others (and you should!), remember to follow the collaboration policy.
In general, you are graded on both the clarity and accuracy of your work. Your solution should be clear enough that someone in the class who had not seen the problem before would understand it.
We sometimes describe approximately how long our explanations are. These are intended to help you understand approximately how much detail we are expecting. Whenever an exercise asks to "*prove*" or "*show*" a statement without further specification, you are free to choose any proof method that you have learned in 311.

Be sure to read the grading guidelines for more information on what we're looking for.

While working on the assignment, please (roughly) keep track of the time you spent on each problem. This is so that you have an idea of how to answer the feedback questions!

1. Collaborators

List any collaborators (i.e., other students) you worked with and which problems you worked on together, or state that you worked alone.

2. Looking back [* points]

This problem is optional.

Learning from individual feedback on your work is valuable; this problem is a structured way to do that. From the following three 311 problems, choose the one you got the lowest score on, as a proportion of the total.

- Homework 3, question 8: Inference Proof with Quantifiers
- Homework 4, question 6: I've never seen such raw power[sets]
- Homework 7, question 4: What doesn't kill you makes you stronger

Review the feedback on the problem, answer the following questions, and revise your solution with fixes to any errors.

(a) Characterize the main issue you had with this problem. Was it, for example, confusion with the concepts the proof is about (power sets, recursion), a logical issue with the structure of the proof, a writing/communication issue (e.g. misused notation), or something else?

(b) What invalid assumptions does your original proof make, if any?

(c) What advice would you give to someone approaching a problem like this for the first time?

Your revised solution will be graded like the original, and the new grade will be used to update your grade *on that earlier assignment*.

3. Formal Proofs and Truth Tables [10 points]

Prove the following using logical equivalences and a truth table. $\neg(p \rightarrow (p \wedge \neg q)) \equiv p \wedge q$.

4. Injecting relations [15 points]

Define the following predicate for binary relations. $\mathsf{inj}(f) := \forall a \forall b \forall c\, ((a,c) \in f \land (b,c) \in f) \to a = b$

Write a formal (predicate logic) proof of $\mathsf{inj}(f \circ g)$, given $\mathsf{inj}(f)$ and $\mathsf{inj}(g)$. In English, this says that the composition of injective relations is injective.

For this proof, you may use the following inference rule, for any variables s and t and any statements P and Q depending on s and t, respectively.

$$\frac{s = t,\ P(s),\ Q(t)}{\therefore\ P(u) \land Q(u),\ u \text{ is fresh}} \quad \text{Leibniz's rule}$$

Recall also the definition of relation composition. If A and B are relations, then $A \circ B$ is defined as $\{(x,z) : \exists y\, (x,y) \in A \land (y,z) \in B\}$. In your proof, you may use this definition to assert $\forall A \forall B \forall x \forall z\, (x,z) \in A \circ B \leftrightarrow \exists y\, ((x,y) \in A \land (y,z) \in B)$.

Errata:

- In this proof you may use $\forall$ introduction/elimination multiple times in a single step.
- This proof is long (ours is ≈30 lines) but only a couple of those lines are the tricky part: don't lose sight of the big picture.
- Our proof contains the following, somewhere in it. Feel free to re-use this however you wish. The LaTeX source of this is available with the template

1. $(i,k) \in f \circ g$ ________
2. $(j,k) \in f \circ g$ ________
3. $\forall A \forall B \forall x \forall z\, (x,z) \in A \circ B \leftrightarrow \exists y\, ((x,y) \in A \land (y,z) \in B)$ Def of composition
4. $(i,k) \in f \circ g \leftrightarrow \exists y\, ((i,y) \in f \land (y,k) \in g)$ $\forall$ elim (4x) from 3
5. $((i,k) \in f \circ g \to \exists y\, ((i,y) \in f \land (y,k) \in g)) \land (\exists y\, ((i,y) \in f \land (y,k) \in g) \to (i,k) \in f \circ g)$ Def of biconditional
6. $(i,k) \in f \circ g \to \exists y\, ((i,y) \in f \land (y,k) \in g)$ $\land$ elimination
7. $\exists y\, ((i,y) \in f \land (y,k) \in g)$ MP 1,6
8. $(i,s) \in f \land (s,k) \in g$ $\exists$ elimination (s is fresh)
9. $(j,k) \in f \circ g \leftrightarrow \exists y\, ((j,y) \in f \land (y,k) \in g)$ $\forall$ elim (4x) from 3
10. $((j,k) \in f \circ g \to \exists y\, ((j,y) \in f \land (y,k) \in g)) \land (\exists y\, ((j,y) \in f \land (y,k) \in g) \to (j,k) \in f \circ g)$ Def of biconditional
11. $(j,k) \in f \circ g \to \exists y\, ((j,y) \in f \land (y,k) \in g)$ $\land$ elimination
12. $\exists y\, ((j,y) \in f \land (y,k) \in g)$ MP 2,11
13. $(j,t) \in f \land (t,k) \in g$ $\exists$ elimination (t is fresh)
14. $(i,s) \in f$ $\land$ elimination 8
15. $(s,k) \in g$ $\land$ elimination 8
16. $(j,t) \in f$ $\land$ elimination 13
17. $(t,k) \in g$ $\land$ elimination 13

5. Extended Euclidean Algorithm [16 points]

Recall that the *Extended Euclidean algorithm* computes the greatest common divisor of two integers $a \geq b \geq 0$ with $(a, b) \neq (0, 0)$ as well as $s, t \in \mathbb{Z}$ so that $\gcd(a, b) = sa + tb$. The algorithm that we saw in the lecture was as follows:

```
Extended Euclidean algorithm: ExtGCD($a, b$) → $(g, s, t)$
Input: Integers $a \geq b \geq 0$ with $(a, b) \neq (0, 0)$
Output: Integers $(g, s, t)$ so that $g = \gcd(a, b)$ and $g = sa + tb$
  (1)  IF $b = 0$ THEN return $(a, 1, 0)$
  (2)  Compute $(g, s, t) :=$ ExtGCD($b, a \% b$)
  (3)  Write $a = qb + (a \% b)$ for some $q \in \mathbb{Z}$
  (4)  Return $(g, t, s - tq)$
```

We can now complete the correctness proof of the algorithm. Let $a, b \in \mathbb{N}$ with $a \geq b \geq 0$ and $(a, b) \neq (0, 0)$.

(a) Prove by strong induction over b that the parameter g which the algorithm returns is indeed equal to $\gcd(a, b)$.

(b) Prove by strong induction over b that the parameters g, s, t which the algorithm returns satisfy $g = sa + tb$.

Hint: Lecture 15 on the Extended Euclidean algorithm may be useful; the most relevant section of the recording may begin around the 12 minute mark.

6. Substitute Teacher [20 points]

When studying modular arithmetic, we made the claim that, if we replace a variable in an arithmetic expression with a value that it is congruent to, the resulting expression will be congruent to the one we started with. We now have sufficient tools to prove that, and we will do so in this problem. We start by formalizing arithmetic expressions. Consider the set **Expr**, representing parse trees for arithmetic expressions, defined recursively as follows:

Bases Step: Variable(x) is in **Expr**, representing the variable x, and for any $v \in \mathbb{Z}$, Number(v) is in **Expr**, representing the integer v.

Recursive Step: For any s and t in **Expr**, Plus(s, t) is in **Expr**, representing $s + t$, and Times(s, t) is in **Expr**, representing $s \cdot t$.

For example, the expression "$3 + 4x$" would become Plus(Number(3), Times(Number(4), Variable(x))). Every possible arithmetic expression (with a single variable x) can be represented by an element of **Expr**. If we choose a specific, integer value $v \in \mathbb{Z}$ for the variable x, then we can calculate the value of the entire expression, with $x = v$, recursively, as follows:

$$\begin{aligned}
\mathsf{value}_v(\mathsf{Number}(w)) &= w && \forall w \in \mathbb{Z} \\
\mathsf{value}_v(\mathsf{Variable}(x)) &= v && \\
\mathsf{value}_v(\mathsf{Plus}(a, b)) &= \mathsf{value}_v(a) + \mathsf{value}_v(b) && \forall a, b \in \textbf{Expr} \\
\mathsf{value}_v(\mathsf{Times}(a, b)) &= \mathsf{value}_v(a) \cdot \mathsf{value}_v(b) && \forall a, b \in \textbf{Expr}
\end{aligned}$$

Now, we define the function subst$_c$, which substitutes the expression $c \in$ **Expr** for all instances of the variable x in an arithmetic expression, resulting in a new **Expr**, as follows:

$$\begin{aligned}
\mathsf{subst}_c(\mathsf{Number}(w)) &= \mathsf{Number}(w) && \forall w \in \mathbb{Z} \\
\mathsf{subst}_c(\mathsf{Variable}(x)) &= c && \\
\mathsf{subst}_c(\mathsf{Plus}(a, b)) &= \mathsf{Plus}(\mathsf{subst}_c(a), \mathsf{subst}_c(b)) && \forall a, b \in \textbf{Expr} \\
\mathsf{subst}_c(\mathsf{Times}(a, b)) &= \mathsf{Times}(\mathsf{subst}_c(a), \mathsf{subst}_c(b)) && \forall a, b \in \textbf{Expr}
\end{aligned}$$

With those definitions in place, we want to prove:

$$\forall v \in \mathbb{Z}\ \Big(\big(\mathsf{value}_v(\mathsf{Variable}(x)) \equiv \mathsf{value}_v(c) (\mathrm{mod}\ m)\big) \to \Big(\forall a \in \textbf{Expr}.\big(\mathsf{value}_v(a) \equiv \mathsf{value}_v(\mathsf{subst}_c(a)) (\mathrm{mod}\ m)\big)\Big)\Big),$$

where $m > 0$ is an integer and $c \in$ **Expr** is a fixed arithmetic expression. In English, this says, for any value we might pick for x, if the value of the expression c is congruent to that of x, the value of any expression a that uses the variable x is congruent to the value of the same expression with c substituted for x. (I.e., if we replace every instance of x in the expression a by c, we get an expression whose value is congruent to that of a.) For example, if c is the **Expr** for $x + 3m$, then the claim says that we can replace x by $x + 3m$ in any **Expr** and the result will be an **Expr** whose value is congruent to the original one, modulo m, since $x \equiv x + 3m \pmod m$. For example, we could replace x by $x + 3m$ in the **Expr** for $5(x + 7)$ to get the **Expr** for $5((x + 3m) + 7)$, which would simplify to $5x + 15m + 35$. This second expression is not the same as the first, but their difference, $15m$, disappears when we work modulo m.

Prove the claim above using structural induction over $a \in$ **Expr**.

Hint: Our solution is shorter than the problem statement! This proof also is all "easy parts". It requires no deep insights. To prove it, just follow the structure of the claim we are trying to prove....

7. Define the Relationship [15 points]

For $m \in \mathbb{N} \setminus \{0\}$, define the relation C_m on $\mathbb{Z}$ by $(a, b) \in C_m$ iff $a \equiv b \pmod m$. State whether each of the following are True or False, and prove your answer.

(a) If $(a, b) \in C_m$ then $(-a, b) \in C_m$.

(b) $C_m^* = C_m$. Recall C_m^* is the reflexive transitive closure of C_m.

(c) State whether or not C_m is reflexive, symmetric, antisymmetric, and transitive. Justify your answer briefly. (You may use any of your previous parts.)

8. Up and Down with Context [Online, 10 points]

Construct a CFG for the language $L \subseteq \{0, 1, 2\}^*$ of all strings that (a) have an equal number of 0's and 1's and (b) where consecutive digits differ by exactly 1. For example 0121010 is in L.

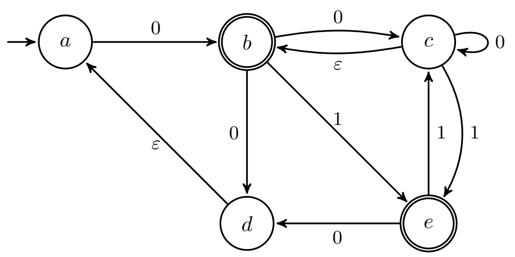
You should submit (and check!) your answers online at https://grin.cs.washington.edu/

Test your grammar carefully before submitting; you only have 5 submissions. Because these are auto-graded, we will not award partial credit.

Hint: As with most problems of this format, it is highly recommended to begin by producing a list of strings in L and a list of strings not in L.

9. Not So Nondeterministic [Online, 10 points]

Use the construction from lecture to convert the following NFA to a DFA.



You should submit (and check!) your answers online at https://grin.cs.washington.edu/

Test your DFA carefully before submitting; you only have 5 submissions. Because these are auto-graded, we will not award partial credit.

10. Regular or Not, That is the Final Question [20 points]

Let 1^* be the regular language of all strings consisting of only ones. Does there exist a set of strings $L \subseteq 1^*$ such that L is an irregular language? Prove that your answer is correct.

11. Pick a card, any card [Extra credit]

Ponder the following thought experiment. Start with a small deck of cards. At each step, remove one card from the bottom and add two new cards to the top. How many cards remain in the deck after infinitely many steps? What if, instead, the card is removed from the top?

To formalize this, we define the sets S and T recursively as follows, numbering each card as it is added to the deck.

(a) Fill in the blank in the recursive step of the following definition so that the set S_{n+1} accurately represents the intermediate state of the deck which has cards removed from the bottom.

- $S_1 = \{1\}$
- $S_{n+1} = (S_n \setminus \underline{\hspace{2cm}}) \cup \{2n, 2n+1\}$
- $S = \{x : \exists m \forall k\, k \geq m \to x \in S_k\}$

(b) Fill in the blank in the recursive step of the following definition so that the set T_{n+1} accurately represents the intermediate state of the deck which has cards removed from the top.

- $T_1 = \{1\}$
- $T_{n+1} = (T_n \setminus \underline{\hspace{2cm}}) \cup \{2n, 2n+1\}$
- $T = \{x : \exists m \forall k\, k \geq m \to x \in T_k\}$

(c) Determine, with proof, whether the set S has infinitely many elements.

(d) Determine, with proof, whether the set T has infinitely many elements.

12. Feedback

Please keep track of how much time you spend on this homework and answer the following questions. This can help us calibrate future iterations of the course, and can help you identify which areas are most challenging for you.

- How many hours did you spend working on this assignment?
- Which problem did you spend the most time on?
- Which problem did you find to be the most confusing?
- Any other feedback for us?