CSE 311: Foundations of Computing I Spring 2024

Problem Set 7

Due: Wednesday, May 22nd by 11:00pm

Instructions

Solutions submission. You must submit your solution via Gradescope. In particular:

- Submit a *single* PDF file containing your solutions to Task 2, 3, 4(b), 5(b), 6 (and optionally 7). Follow the prompt on Gradescope to link tasks to your pages.
- The instructions for submitting Tasks 1, 2, 4(a), and 5(a) appear below those individual problems.

Task 1 – Hope Strings Eternal [10 pts]

For each of the following, construct regular expressions that match the given set of strings:

a) Binary strings that start with 1 and have odd length.

b) Binary strings where every 1 is immediately followed by a 00.

c) Binary strings with an odd number of 1s.

d) Binary strings with at least three 1s.

e) Binary strings with at least three 1s **or** at most two 0s.

> Submit and check your answers to this question here:
>
> `http://grin.cs.washington.edu`
>
> Think carefully about your answer to make sure it is correct before submitting. You have only **5 chances** to submit a correct answer.

Task 2 – Glitz and Grammar [15 pts]

For each of the following, a construct context-free grammar that generates the given set of strings.

If your grammar has more than one variable, write a sentence describing what sets of strings you expect each variable in your grammar to generate. For example, if your grammar was:

$$\mathbf{S} \to \mathbf{E} \mid \mathbf{O}$$
$$\mathbf{O} \to \mathbf{EC}$$
$$\mathbf{E} \to \mathbf{EE} \mid \mathbf{CC}$$
$$\mathbf{C} \to 0 \mid 1$$

You could say "C generates binary strings of length one, E generates (non-empty) even length binary strings, and O generates odd length binary strings." It is also fine to use a regular expression, rather than English, to describe the strings generated by a variable (assuming such a regular expression exists).

a) Binary strings matching the regular expression "$0(1 \cup 00)^* \cup 111$".

Hint: You can use the technique described in lecture to convert this RE to a CFG.

b) All strings of the form $x\#y$, with $x, y \in \{0,1\}^*$ and y a subsequence of x^R.

(Here x^R means the reverse of x. Also, a string w is a subsequence of another string z if you can delete some characters from z to arrive at w.)

c) All binary strings in the set $\{0^m 1^n : m, n \in \mathbb{N} \text{ and } m < n\}$.

> Submit and check your answers to this question here:
>
> http://grin.cs.washington.edu
>
> Think carefully about your answer to make sure it is correct before submitting. You have only **5 chances** to submit a correct answer.
>
> **Note**: You must also include each grammar and sentences describing each new non-terminal, as described above, in Gradescope.

Task 3 – 101 Relations [12 pts]

For each of the relations below, determine whether or not it has each of the properties of reflexivity, symmetry, antisymmetry, and/or transitivity. If a relation has a property, simply say so without any further explanation. If a relation does not have a property, state a counterexample, but do not explain your counterexample further.

a) Define $R \subseteq \mathbb{Z} \times \mathbb{Z}$ by $(a, b) \in R$ iff ($a = b$ or $a = -b$).

b) Define $S \subseteq \mathbb{Z} \times \mathbb{Z}$ by $(a, b) \in S$ iff $|a - b| \leq 1$ (where $|x|$ denotes the absolute value of x, i.e., x if x is nonnegative, and $-x$ if x is negative).

c) Define $T \subseteq \mathbb{Z} \times \mathbb{Z}$ by $(a, b) \in T$ iff $a \not\equiv_5 b$.

d) Let $A = \{n \in \mathbb{N} : n > 0\}$ be the set of positive natural numbers. Define $U \subseteq A \times A$ by $(a, b) \in U$ iff $a \mid b$, i.e., a divides b.

e) Let $B = \mathcal{P}(\mathbb{Z})$. Define $V \subseteq B \times B$ by $(X, Y) \in V$ iff $X \cap [10] \subseteq Y \cap [10]$. (Remember that $[n] = \{1, \ldots, n\}$.)

f) Let $A = \{n \in \mathbb{N} : n > 0\}$ be the set of positive natural numbers. Define $W \subseteq A \times A$ by $(a, b) \in W$ iff $a \mid 3$ and $b \mid 2$ and ($a = 1$ or $b = 1$).

Hint: Be careful! The definition of W says $a \mid 3$, which is (very) different from $3 \mid a$.

Task 4 – Get a Prove On [16 pts]

Let R and S be relations on a set A. Consider the following claim:

Given that R and S are transitive, it follows that $R \cap S$ is transitive.

a) Write a **formal proof** that the claim holds.

In Cozy's syntax, you write R is transitive as "`Transitive(R)`", and you write $(a, b) \in R$ as "`pair(a, b) in R`".

Your proof will need the definition of transitive in addition to the other set operations. In Cozy, you can replace "`Transitive(R)`" by its definition with "`defof Transitive`" and the reverse with "`undef Transitive`". For this problem, you can also use "`defof Intersection`" rather than "`defof cap {R} {S}`" to replace "$x \in R \cap S$" with its definition.

> Submit and check your formal proof here:
>
> http://cozy.cs.washington.edu
>
> You **must also** include your solution (as a screenshot, typeset LaTeX, or rewritten by hand) in the PDF you submit to **Gradescope**.

b) Translate your formal proof to an **English proof**.

Keep in mind that your proof will be read by a *human*, not a computer, so there are many formal steps you can skip. Your English proof should be much shorter than your formal proof.

Task 5 – Cat Goes "Meow". Dog Goes "Proof" [16 pts]

Let R and S be relations on a set A. Consider the following claim:

Given that R and S are symmetric and $R \circ S = S \circ R$, it follows that $R \circ S$ is symmetric.

a) Write a **formal proof** that the claim holds.

In Cozy's syntax, you write R is symmetric as "`Symmetric(R)`", you write the set $R \circ S$ as "`compose(R, S)`", and you write $R \circ S = S \circ R$ as "`compose(R, S) sameset compose(S, R)`".

Your proof will need the definitions of symmetric and composition in addition to the other set operations. In Cozy, you can replace "`Symmetric(R)`" by its definition with "`defof Symmetric`" and the reverse with "`undef Symmetric`", and you can replace "$(x, y) \in R \circ S$" by its definition with "`defof Compose`" and the reverse with "`undef Compose`".

> Submit and check your formal proof here:
>
> http://cozy.cs.washington.edu
>
> You **must also** include your solution (as a screenshot, typeset LaTeX, or rewritten by hand) in the PDF you submit to **Gradescope**.

b) Translate your formal proof to an **English proof**.

Keep in mind that your proof will be read by a *human*, not a computer, so there are many formal steps you can skip. Your English proof should be much shorter than your formal proof.

Task 6 – Long Live the String [44 pts]

In section, we defined the set of binary strings of even length recursively as follows:

Basis: $\varepsilon \in S$.
Recursive Step: If $x \in S$ and $a, b \in \{0,1\}$, then $xab \in S$.

We could instead define the set of binary strings of even length more directly like this:

$$T := \{x \in \{0,1\}^* : 2 \mid \mathsf{len}(x)\}$$

In the following parts, we will prove that the two definitions are equivalent, i.e., that $S = T$.

a) Use structural induction to prove that $\forall x \in S\,(x \in T)$.

b) Use strong induction to prove that $\forall n \in \mathbb{N}\,(\forall x \in \{0,1\}^*(\mathsf{len}(x) = 2n \to x \in S))$.

You will need to use the fact that, if $\mathsf{len}(x) = 0$, then $x = \varepsilon$, which we will call Lemma 1, and the fact that, if $\mathsf{len}(x) > 0$, then $x = ya$ for some $a \in \{0,1\}$ and $y \in \{0,1\}^*$, which is Lemma 2.

c) Explain, in at most three sentences, why parts (a-b) tell us that $S = T$.

Task 7 – Extra Credit: With a Grammar, the Whole World is a Nail [0 pts]

Consider the following context-free grammar.

$$
\begin{array}{ll}
\langle \textbf{Stmt} \rangle & \to \langle \textbf{Assign} \rangle \mid \langle \textbf{IfThen} \rangle \mid \langle \textbf{IfThenElse} \rangle \mid \langle \textbf{BeginEnd} \rangle \\
\langle \textbf{IfThen} \rangle & \to \text{if condition then } \langle \textbf{Stmt} \rangle \\
\langle \textbf{IfThenElse} \rangle & \to \text{if condition then } \langle \textbf{Stmt} \rangle \text{ else } \langle \textbf{Stmt} \rangle \\
\langle \textbf{BeginEnd} \rangle & \to \text{begin } \langle \textbf{StmtList} \rangle \text{ end} \\
\langle \textbf{StmtList} \rangle & \to \langle \textbf{StmtList} \rangle \langle \textbf{Stmt} \rangle \mid \langle \textbf{Stmt} \rangle \\
\langle \textbf{Assign} \rangle & \to \texttt{a := 1}
\end{array}
$$

This is a natural-looking grammar for part of a programming language, but unfortunately the grammar is "ambiguous" in the sense that it can be parsed in different ways (that have distinct meanings).

a) Show an example of a string in the language that has two different parse trees that are meaningfully different (i.e., they represent programs that would behave differently when executed).

b) Give **two different grammars** for this language that are both unambiguous but produce different parse trees from each other.