CSE 311: Foundations of Computing I Winter 2025

Problem Set 8

Due: Thursday, March 13th by 11:00pm

Instructions

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

- Submit a *single* PDF file containing your solutions to Tasks 1, 2, 5, 6, 7 (and optionally 8). Follow the prompt on Gradescope to link tasks to your pages.
- Tasks 3 & 4 should be submitted **only** on `grin`. The instructions for appear below that problem.
- Task 6 should be submitted **first** on `grin`. However, a screenshot of your DFA also need to be submitted as part of your PDF in Gradescope so that we can verify that the states are named by subsets of states of the NFA.

Task 1 – 101 Relations [10 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 $a - b$ is even.

b) Define $S \subseteq \mathbb{Z} \times \mathbb{Z}$ by $(a, b) \in S$ iff $a + b = 0$.

c) Let $A = \{n \in \mathbb{N} : n > 0\}$ be the set of positive natural numbers. Define $T \subseteq A \times A$ by $(a, b) \in T$ iff $2a \mid b$, i.e., $2a$ divides b.

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

e) Let $A = \{n \in \mathbb{N} : n > 0\}$ be the set of positive natural numbers. Define $V \subseteq A \times A$ by $(a, b) \in V$ iff $a \mid 5$ and $b \mid 4$ and ($a = 1$ or $b = 1$).

Hint: Be careful! The definition of V says $a \mid 5$, which is (very) different from $5 \mid a$.

Task 2 – Better Get Proving [10 pts]

Let R be a relation on a set A. Recall from lecture that "R is *symmetric*" iff

$$\forall a \, \forall b \, ((a, b) \in R \rightarrow (b, a) \in R)$$

i.e. if a pair (a, b) is in R, then swapped pair (b, a) is also in R.

Also, recall the definition of the *composition* of two relations from lecture. In particular,

$$R \circ R := \{(a, b) \mid \exists \mathbf{c} \, ((a, \mathbf{c}) \in R \wedge (\mathbf{c}, b) \in R)\}$$

Write an **English** proof of the following claim: if R is symmetric, then $R \circ R$ is symmetric.

Task 3 – Few and Far Machine [25 pts]

For each of the following, create a *DFA* that recognizes exactly the language given.

a) Binary strings that start with 10 and have an even length.

b) Binary strings where every 1 is followed by an even number of 0s. That is, if s contains a 1, meaning it can be written as $s = x1 \bullet y$, for some strings x, y, then y *must* have an even number of 0s.

c) Binary strings with an odd number of 1s where every 1 is immediately followed by a 0.

d) Binary strings with at most three 1s.

e) Binary strings with at most three 1s **or** at least 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 4 – Design Intervention [15 pts]

For each of the following, create an *NFA* that recognizes exactly the language described.

a) Binary strings with at least three 1s **or** at least three 0s.

Hint: Since this is an NFA, your answer can be much smaller than for Task 3(e).

b) Binary strings that end in 111 but do not contain the substring 00.

c) Binary strings that start with 10, end in 01, **and** contain at most three 0s in total. Notice that 101 is in the language.

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 5 – The Great Expression [10 pts]

Use the algorithm from lecture to convert each of the following regular expressions into NFAs that accept the same language. You may skip adding ε-transitions for concatenation if they are *obviously* unnecessary, but otherwise, you should **precisely** follow the construction from lecture.

a) $1(0 \cup 10)^*0 \cup 001$

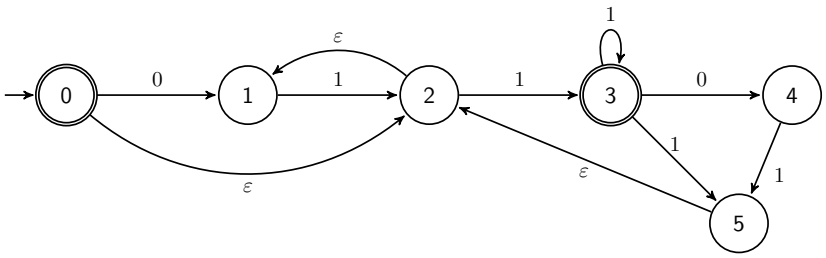
(Note: We translated this into a CFG in HW7 Task 6(a). Here, we translate it into an NFA.)

b) $(00(11 \cup 0)^*)^*$

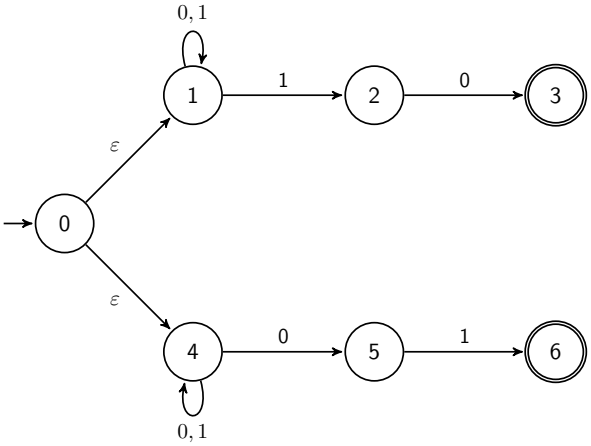
Task 6 – Machine With Envy [10 pts]

Use the algorithm from lecture to convert each of the following NFAs to DFAs. Label each DFA state with the set of NFA states it represents in the powerset construction.

a) The NFA below, which accepts the empty string and *some* strings ending with 1:



b) The NFA below, which accepts strings ending in "10" or "01":



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 a screenshot of each DFA in your submission to Gradescope so that we can verify that you properly named each of the states as a subset of the NFA states.

Task 7 – Just Irregular Guy [20 pts]

Use the method described in lecture to prove that each of the following languages is **not regular**.

(Note: we saw that *there are* CFGs for both of the languages below in HW7 Task 6.)

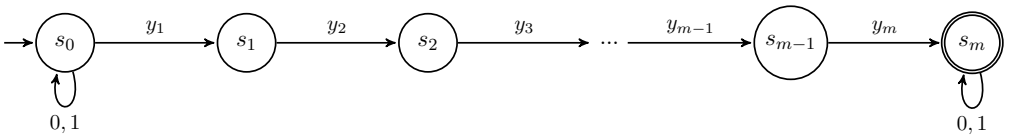
a) All binary strings in the set $\{0^m 1^n : m, n \in \mathbb{N} \text{ and } m > n\}$.

b) All strings of the form $y\#x$, 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.)

Task 8 – Extra Credit: Strings to Mind [0 pts]

Suppose we want to determine whether a string x of length n contains a string $y = y_1 y_2 \dots y_m$ with $m \ll n$. To do so, we construct the following NFA:



(where the ... includes states $s_3, \dots, s_{m-2}$). We can see that this NFA matches x iff x contains the string y.

We could check whether this NFA matches x using the parallel exploration approach, but doing so would take $O(mn)$ time, no better than the obvious brute-force approach for checking if x contains y. Alternatively, we can convert the NFA to a DFA and then run the DFA on the string x. *A priori*, the number of states in the resulting DFA could be as large as 2^m, giving an $\Omega(2^m + n)$ time algorithm, which is unacceptably slow. However, below, you will show that this approach can be made to run in $O(m^2 + n)$ time.

(a) Consider any subset of states, S, found while converting the NFA above into a DFA. Prove that, for each $1 \leqslant j < m$, knowing $s_j \in S$ *functionally determines* whether $s_i \in S$ or not for each $1 \leqslant i < j$.

(b) Explain why this means that the number of subsets produced in the construction is at most $2m$.

(c) Explain why the subset construction thus runs in only $O(m^2)$ time (assuming the alphabet size is $O(1)$).

(d) How many states would this reduce to if we then applied the state minimization algorithm?

(e) Explain why part (c) leads to a bound of $O(m^2 + n)$ for the full algorithm (without state minimization).

(f) Briefly explain how this approach can be modified to count (or, better yet, find) *all* the substrings matching y in the string x with the same overall time bound.

Note that any string matching algorithm takes $\Omega(m + n) = \Omega(n)$ time in the worst case since it must read the entire input. Thus, the above algorithm is optimal whenever $m^2 = O(n)$, or equivalently, $m = O(\sqrt{n})$, which is the case for normal inputs circumstances.