Sample Solutions
CS314 Spring 2023 Homework 1

Problem 1 - 15 pts

1.1 Successor function
Rule 1: $\$| \Rightarrow |\$$
Rule 2: $\$\# \Rightarrow |$

1.2 Triple function
Rule 1: $\$| \Rightarrow |||\$$
Rule 2: $\$\# \Rightarrow \epsilon$

1.3 Subtraction function
Rule 1: $\$ \Rightarrow \epsilon$
Rule 2: $\&\# \Rightarrow \epsilon$
Rule 3: $|\&| \Rightarrow \&$

Problem 2 - 15 pts

Rule 1: $(0 + 0) \Rightarrow 0$
Rule 2: $(0 + 1) \Rightarrow 1$
Rule 3: $(1 + 0) \Rightarrow 1$
Rule 4: $(0 + 2) \Rightarrow 2$
Rule 5: $(2 + 0) \Rightarrow 2$
Rule 6: $(1 + 1) \Rightarrow 2$
Rule 7: $(1 + 2) \Rightarrow 0$
Rule 8: $(2 + 1) \Rightarrow 0$
Rule 9: $(2 + 2) \Rightarrow 1$

Example : $\Big(\big(1 + 2\big) + 0\Big)$
$\Big(\big(1 + 2\big) + 0\Big) \Rightarrow \Big(0 + 0\Big)$ (using Rule 7)
$\Big(0 + 0\Big) \Rightarrow 0$ (using Rule 1)

Example : $\Big(1 + \big(2 + 2\big)\Big)$

$\Big(1 + \big(2 + 2\big)\Big) \Rightarrow \Big(1 + 1\Big)$ (using Rule 9)

$\Big(1 + 1\Big) \Rightarrow 2$ (using Rule 6)

The choice of the next rewrite rule and its left hand side is not always unique in this rewrite system. Consider the following example where you can apply either rule 6 or rule 8.

Example: $\Big(\big(1 + 1\big) + \big(2 + 1\big)\Big)$

Problem 3 - 10 pts

$((\epsilon \mid 1)0^*)^*$

All strings over the alphabet {0,1}, including ϵ

$0(0 \mid 1)^*1(0 \mid 1)1$

All binary strings that start with 0 and end with 101 or 111.

Problem 4 - 15 pts

1. All strings of "a"s, "b"s, and "c"s that contain exactly 2 "a"s

   $(b \mid c)^*\ a\ (b \mid c)^*\ a\ (b \mid c)^*$

2. All strings of "a"s, "b"s, and "c"s that contain at least 1 "b" or at least 3 "c"s.

   $(a \mid b \mid c)^*\ b\ (a \mid b \mid c)^*\ \mid\ (a \mid b \mid c)^*\ c\ (a \mid b \mid c)^*\ c\ (a \mid b \mid c)^*\ c\ (a \mid b \mid c)^*$

3. All strings of "a"s, "b"s, and "c"s that do not contain more than 1 "b" and no more than 3 "c"s.

   $$\begin{array}{ll} & a^*((b \mid \epsilon)a^*(c \mid \epsilon)a^*(c \mid \epsilon)a^*(c \mid \epsilon)a^* \\ \mid & a^*((c \mid \epsilon)a^*(b \mid \epsilon)a^*(c \mid \epsilon)a^*(c \mid \epsilon)a^* \\ \mid & a^*((c \mid \epsilon)a^*(c \mid \epsilon)a^*(b \mid \epsilon)a^*(c \mid \epsilon)a^* \\ \mid & a^*((c \mid \epsilon)a^*(c \mid \epsilon)a^*(c \mid \epsilon)a^*(b \mid \epsilon)a^* \end{array}$$

Problem 5 - 30 pts

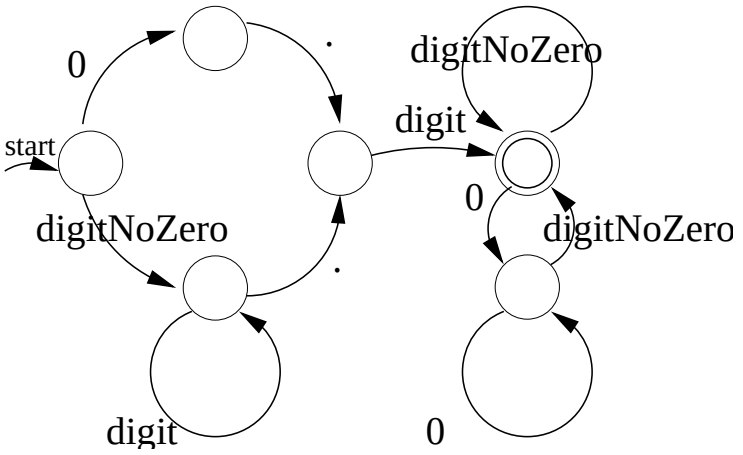
You are designing a new language with fixed-point numbers. Every fixed-point number should have a unique representation. This means, no leading or trailing 0's are allowed, and every number must have a "point".

- Allowed: `0.0`, `10.0`, `45000.007`, `0.888`
- Not allowed: `0`, `10`, `10.`, `10.00`, `045000.007`, `.888`

1. Write a regular expression for fixed-point numbers for your language. Define the following macros:
   $digit = (0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9)$
   $digitNoZero = (1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9)$

   `Answer`: $(0 \mid (digitNoZero \;\; digit^*)).(0 \mid (digit^* \;\; digitNoZero))$

2. Give a DFA in the form of a state transition graph that recognizes your language. Note: No need to introduce error states; your DFA can reject the input if it gets "stuck". Keep your DFA as small as possible.



Problem 6 - 15 pts

Use the discussed "translation" strategy for constructing an ϵ-NFA from a regular expression as discussed in lecture 3 for the regular expression.

letter (letter | digit)*

Show the ϵ-NFA for the above regular expression.