Programming Languages and Computation

Week 2: Regular Expressions

* 1. For each of the following regular expressions over $\{0, 1\}$, write it out in the official syntax with all of the implicit parentheses and the concatenation operators tediously put back in (according to our agreed conventions).

(a) $(00^* + 11^*)^*$

(b) $(01 + 10)(01 + 10)$

(c) $(0 + 1(01^*0)^*1)^*$

* 2. For each of the following regular expressions over $\{a, b\}$, write it out with the minimum number of parentheses and all concatenation operators suppressed (according to our agreed conventions).

(a) $(((a + \epsilon) \cdot (b + \epsilon))^*)$

(b) $(((a + b) \cdot (b^*))^*)$

(c) $(((a^*) + (((a \cdot b) \cdot a)^*))^*)$

* 3. Consider the following pairs of regular expressions over $\{a, b\}$. For each, find a word that is matched by the first, but not by the second.

(a) $(a + b)^*$ $a^* + b^*$

(b) $(a^*b)^*$ $(ab^*)^*$

(c) $\emptyset^*$ $(a + b)$

* 4. Give traces to show that the regular expression $(ab + a)^*$ matches each of the following strings:

(a) ϵ

(b) aab

(c) $abab$

** 5.

(a) Draw a directed graph with the following components:

- The nodes/vertices are: $\{R \mid (a^* + b)^* \rightarrow^* R\}$
- There is an ℓ-labelled edge between vertex R and vertex S just if $R \xrightarrow{\ell} S$.

(b) Give infinitely many different traces to show that the regular expression $(a^* + b)^*$ matches the string ab.

(You do not have enough paper to enumerate them explicitly, so please just indicate the general pattern.)

** 6. Design regular expressions to match the following strings over $\{a, b\}$:

(a) All strings that contain an even number of a

(b) All strings that contain an odd number of b

(c) All strings that contain either an even number of a or an odd number of b

** 7. Design regular expressions for the following programming language lexemes over the ASCII alphabet. You will find it convenient to define abbreviations to help present the expressions compactly.

(a) A *C program identifier* is any string of length at least 1 containing only letters ('a'–'z', lower and uppercase), digits ('0'–'9') and the underscore, and which begins with a letter or the underscore.

(b) In most programming languages, *integer literals* can be written in:

- decimal – as a non-empty sequence of digits
- hexadecimal – as a non-empty sequence of characters from '0'–'9','a'–'e' (upper or lowercase) that are preceded by "0x"
- binary – as a non-empty sequence of bits '0' and '1' that are preceded by "0b"

** 8. Design regular expressions to match the following sets of strings over the alphabet of ASCII characters. You will find it convenient to define some abbreviations to help present the expressions compactly.

(a) Valid Bristol University usernames

(b) Valid 24 hour clock times in format HH:MM

(c) Valid IPv4 addresses written in decimal

*** 9. Use traces to argue that $(a^*b^*)^*$ can match all strings over $\{a, b\}$.

Hint: Consider what can be matched using a trace that both starts and ends in $(a^*b^*)^*$.

*** 10. Give regular expressions to match each of the following sets of strings over $\{a, b\}$:

(a) Strings that do not contain the substring a

(b) Strings that do not contain the substring ab

(c) Strings that do not contain the substring aba