CPSC 229, Fall 2010 — Third Test Info

The third test for this course will be given in class on Friday, November 19. (It was originally scheduled for Wednesday, but we agreed to move it.) The test will cover Chapter 3, Section 2 through Chapter 4, Section 1. However, it will ***not*** *include any questions about regular expressions on the computer (Section 3.2). And of course you will need to be very familiar with the basic material on languages from Section 3.1.*

The format will be similar to previous tests. You can expect to apply the NFA-to-DFA conversion algorithm. There might also be a problem using the regular-expression-to-NFA algorithm. Although you will not have to give any detailed mathematical proofs, you might be asked to give the general idea of the proofs of certain facts, such as the fact that the intersection of two regular languages is regular

Here are some terms and ideas that you should be familiar with for the test:

regular expression over an alphabet Σ

ε as a regular expression

the operators $*$, $+$, and concatenation ($\cdot$) in regular expressions

how $L(r)$ is constructed from r

pattern matching; what it means for a string to match a regular expression

the language generated by a regular expression, denoted $L(r)$ for a regular expression r

regular language

FSA (Finite-State Automaton) — these is just a general term for NFA or DFA

DFA (Deterministic Finite Automaton)

transition diagram [the usual picture] of a DFA

state (in a finite-state automaton)

start state

accepting state (also known as final state)

definition of a DFA as a list of five things, $(Q, \Sigma, q_o, \delta, F)$ — and what each thing means

transition function, $\delta\colon Q \times \Sigma \to Q$, of a DFA

transition table for a DFA

the function $\delta^*\colon Q \times \Sigma^* \to Q$

how a DFA computes (that is, what it does when it reads and processes a string)

what it means for a DFA to accept a language

the language, $L(M)$, accepted by a DFA, M; $L(M) = \{w \in \Sigma^* \mid \delta^*(q_o, w) \in F\}$

NFA (Non-deterministic Finite Automaton)

nondeterminism

the differences between NFAs and DFAs

ε-transitions

what it means for an NFA to accept a string

the language, $L(M)$, accepted by an NFA, M
algorithm for converting an NFA to an equivalent DFA
algorithm for converting a regular expression to an equivalent NFA
DFAs, NFAs, and regular expressions define exactly the same class of languages
operations ($L_1 \cup L_2$, $L_1 \cap L_2$, LM, L^R, L^*) on regular languages produce regular languages
idea of proof that the complement of a regular language is regular
idea of proof that the intersection of two regular languages is regular
the pumping lemma
using the pumping lemma to show that certain specific languages are not regular
CFGs (Context-Free Grammars)
production rules
non-terminal and terminal symbols
start symbol
definition of a CFG as a 4-tuple $G = (V, \Sigma, P, S)$
the relations $\Longrightarrow$ and $\Longrightarrow^*$
derivation
the language generated by a CFG, $L(G) = \{\, w \in \Sigma^* \mid S \Longrightarrow^* w \,\}$
context-free language
finding a CFG for a given language and *vice versa*
if L and M are context-free languages, then so are $L \cup M$, LM, and L^*
every regular language is context-free
examples of languages that are not regular, such as:

$\{a^n b^n \mid n \in \mathbb{N}\}$
$\{w \in \{a, b\}^* \mid w = w^R\}$
$\{w \in \{a, b\}^* \mid n_a(w) < n_b(w)\}$
$\{a^{n^2} \mid n \in \mathbb{N}\}$
$\{ww \mid w \in \{a, b\}^*\}$

examples of languages that are context-free but not regular, such as:

the first three languages in the previous list

some tasks that you can be asked to perform:

giving an English description of a language
finding a regular expression for a given language
finding a DFA or NFA for a given language
finding a regular expression for an NFA or DFA
finding a CFG for a given language
finding the language generated by a give CFG