<u>Lecture 8: 4/23/2009</u>

Announcements: Midterm Tues. 5/5: Open book/notes
Ps4 out,Ps3 solutions out;

Formal Languages (Chapter 12): This topic is discussed in much greater detail in ECS120

Regular Expressions, continued:12.4

Example 1: write a regular expression for all strings over {a,b} whose length is divisible by 3.

((aub)(aub)(aub))*

Example 2: write a regular expression for all strings over {a,b} whose length is NOT divisible by 3.

(aub)(aub)(aub))*(aub) u
(aub)(aub)(aub))*(aub)(aub)

Example 3: write a regular expression for all strings over {0,1}that contain _some_ '111'.

(0u1)* 111 (0u1)*

Exercise 4: write a regular expression for all strings over {0,1} that contain an even # of 0's and an even # of 1's.

Hmm...... sounds HARD! Can it be done?!
Suggestion in class to take the union of all strings of length 0,2 and 4 with an even number of 0's, 1's, and then take the * of that. However, doesn't quite work.

----
2. FSAs (or DFA, DFM, FSM)(12.5)

Describe pictorially (no actual definition). Machine for each of the 4 examples.

What they capture: a computer with a finite amount of memory.
For a DFM (Will see DFM, DFA, FSA, FSM where D= deterministic, F= finite, A = automaton, M = Machine, S = state)
Make a DFA for each of the languages above.

For an FSA M, L(M) is the **language accepted by M.** A string **x** ∈ L(M) iff on input string **x**, the machine M ends up in an accepting state.

Theorem (ecs 120): L is the language of a regular expression
iff
L is the language of a DFA

So now we _know_ that example 4 does have a regular expression, just our own stupidity not to be able to write it.

Nice new ways to describe languages: give a DFA.

----
3. Relations (chapter 2)

(Change of topics. But do define some relations on strings, regular languages, and DFAs to tie the two topics together.)

DEF: A and B sets. Then a ***relation*** *R* is subset of A x B. Thus R is a set of ordered pairs.

R ⊆ A x B

Variant notation: x R y for (x,y) ∈ R

May use a symbol like ~ or < for a relations

$x \sim y$ if $(x,y) \in \sim$

Relations in arithmetic, where A and B are both natural numbers:

= < ≤ > >=

| divides

what about succ, +, * NO: function symbols, not relations (well, succ *could* be considered a relation: {(0,1), (1,2), …}

In set theory:

∈, =, ⊆ all are relations (on what types?)

what about ∅ NO: constant symbol

Relations are useful for things other than numbers and sets and the like:

S = all UCD students for S09
C = all UCD classes for S09
P = all UCD professors for S09

E: "enrolled in" relation ⊆ S x C
s E c (ie, (s,c)∈ E) - s is taking class c

T: teaches relation ⊆ C x P
c T p (ie, (c,p)∈ T) - professor p is teaching class c this term

Relations of the sort above ("Teaches", "Enrolled" often used in databases, **Relational Databases** store their data as relations)

You can ***compose*** relations (2.5)

what _should_
E o T Mean?

$E \circ T \subseteq S \times P$ $S \times C$ $C \times P \to S \times P$

s EoT p if there exists c in C such that s E c and c T p --
student s is taking some course that p is teaching --
p is s's teacher this term

What I've just given is the general definition
$R \subseteq X \times Y$
$S \subseteq Y \times Z$ then $R \circ S \subseteq X \times Z$ is $\{(x,z): \exists y \text{ in } Y\ xRy \text{ and } yRz\}$

DRAW A GRAPH -- and give the directed-graph interpretation of relations and composition (2.4 in text). Composing relations is called a **Join** operation in a database. (Example below, not done in class)

Students	Classes	Instructors
John	ECS20	Prof. Mar
Nina	Math118	Prof. Hamann
Till	ECS20	Prof. Morrison
Elizabeth	Lin 10	Prof. Samuelson

What _should_ E^{-1} be?
Formalize

if $R \subseteq X \times Y$ is a relation then R^{-1} is the relation on $Y \times X$
where $(y,x) \in R^{-1}$ iff x R y.

More examples:
Often X = Y is the *same* set
Relations on natural numbers, real numbers, strings, etc.

X = set of strings

x <= y "is a substring of y"

alpha and beta are regular expressions.
alpha ~ beta if L(alpha) = L(beta)

T/F: (0u1)* (0u1)* ~ (0 u 1 u 00 u 01 u 10 u 11)* TRUE
emptystring ~ emptystring* TRUE
0(0u1)0 ~ 1(0u1)1 FALSE

M and M' are DFAs.
M ~ M' if L(M) = L(M')

----
4. **Properties of Relations** (2.6)

Reflexive: x R x for all x
Symmetric: x R y -> y R x for all x,y
Transitive: x R y and y R x -> x R z for all x, y, z

if R has all three properties, R is said to be an **equivalence** relation (2.8)

	Reflexive	Symmetric	Transitive	comments
= on Integers (or anything else)	Yes	Yes	Yes	
≤ , integers	Yes	No	Yes	actually _anti-symmetric_ (Define this)
⊆ , sets	Yes	No	Yes	anti-symmetric

x E y if x,y RE L(x) = L(y) languages	Yes	Yes	Yes
x S y if x is a substring of y	Yes	No	Yes
x R y where x and y are strings and M is a some DFA and you go to the same state on processing x and y	Yes	Yes	Yes