T2: mailed back - adjustment: Q1 + 6/13 * (Q2 + Q3)
A3: Questions?
office hours: tomorrow CANCELLED BUT xtra thurs + Fri for 2 weeks...

CSC236 fall 2016

regular expressions

Danny Heap
heap@cs.toronto.edu / BA4270 (behind elevators)
http://www.cdf.toronto.edu/~csc236h/fall/
416-978-5899

Using Introduction to the Theory of Computation, Chapter 7

Outline

regular expressions

NFSAs

regular languages

notes

another way to define languages

In addition to the language accepted by DFSA $L(M)$
and set description $L = \{\ldots\}$.

Definition: The regular expressions (regexps or REs) over alphabet Σ is the **smallest** set such that:

1. $\{\}$, ϵ, and x, for every $x \in \Sigma$ are REs over Σ
2. if T and S are REs over Σ, then so are:
   - $T + S$ (union) — lowest precedence operator
   - TS (concatenation) — middle precedence operator
   - T^* (star) — highest precedence

formally $(T+S) \equiv T+S$

$(TS) \equiv TS$

$(T)^* \equiv T^*$

$R + ST^* \equiv (R + (S(T)^*))$

Computer Science
UNIVERSITY OF TORONTO

regular expression to language:

The $L(R)$, the language denoted (or described) by R is defined by structural induction:

Basis: If R is a regular expression by the basis of the definition of regular expressions, then define $L(R)$:

- $L(\emptyset) = \emptyset$ (the empty language — no strings!)
- $L(\varepsilon) = \{\varepsilon\}$ (the language consisting of just the empty string)
- $L(x) = \{x\}$ (the language consisting of the one-symbol string)

Induction step: If R is a regular expression by the induction step of the definition, then define $L(R)$:

- $L(S + T) = L(S) \cup L(T)$
- $L(ST) = L(S)L(T)$
- $L(T^*) = L(T)^*$

Computer Science
UNIVERSITY OF TORONTO

regexp examples

- java REs
- command-line REs
- $L(0+1) = \{0,1\}$ $= L(0) \cup L(1)$ $L((0^*1^*)^*)$ $\{00, 11, 10, 01\}$ $L(\varepsilon + 0 + 1)$
- $L((0+1)^*)$ All binary strings over $\{0,1\}$
- $L((01)^*) = \{\varepsilon, 01, 0101, 010101, \ldots\}$ $L(01)$ $=$
- $L(0^*1^*)$ 0 or more 0s followed by 0 or more 1s. $\{\varepsilon, 0, 00, 000, 01, 011, 0111, \ldots, 1, 11, 111$
- $L(0^* + 1^*)$ 0 or more 0s or 0 or more 1s.
- $L((0+1)(0+1)^*)$ Non-empty binary strings over $\{0,1\}$.

$L(0+1)\,L(0+1)^* = \{s \in \{0,1\}^* \mid s \text{ begins with } 0 \text{ or } s \text{ begins with } 1\}$

Computer Science
UNIVERSITY OF TORONTO

example

$L' = \{x \in \{0,1\}^* \mid x \text{ begins and ends with a different bit}\}$

$L(0)L(0+1)^*L(1) \cup L(1)L(1+0)^*L(0)$

$R = 0(0+1)^*1 + 1(1+0)^*0$

$L(0+1) = L(0) \cup L(1)$

$= L(1) \cup L(0)$

$= L(1+0)$

Prove $L' = L(R)$

$\Rightarrow$ (Prove $L(R) \subseteq L'$).

Let s be an arbitrary string in $L(R)$.

Then $s \in L(0)L(0+1)^*L(1) \cup L(1)L(0+1)^*L(0)$

Case 1 $s \in L(0)L(0+1)^*L(1)$. Then $s = tuv$, where $t \in L(0)$, $u \in L(0+1)^*$, $v \in L(1)$. Then t (and hence s) begins with 0 and v (and hence s) ends with 1, and $0 \neq 1$

Case 2 Same as case 1 after interchanging 0s and 1s so s begins and ends with different bit

In every possible case, s begins and ends with a different bit, so $s \in L'$. So $L(R) \subseteq L'$, since s is arbitrary

Computer Science
UNIVERSITY OF TORONTO

example

$L' = \{x \in \{0,1\}^* \mid x \text{ begins and ends with a different bit}\}$

(Proof $\Leftarrow$) Must show $L' \subseteq L(R)$.

Let s be an arbitrary string from L'. Then, either s begins with 0 and ends with 1 or s begins with 1 and ends with 0.

WLOG s begins with 0, has zero or more characters after that, and ends with 1. So $s = tuv$ where $t = 0$, u is any binary string, and $v = 1$. that is $t \in L(0)$, $v \in L(1)$ and u consists of 0 or more concatenations from $L(0+1)$. So $s = tuv \in L(0)L(0+1)^*L(1)$

i.e. $s \in L(0(0+1)^*1)$

OR (WLOG) $s \in L(1(0+1)^*0)$

$\Rightarrow s \in L(0(0+1)^*1) \cup L(1(0+1)^*0) = L(0(0+1)^*1 + 1(0+1)^*0)$

$\Rightarrow L' \subseteq L(0(0+1)^*1 + 1(0+1)^*0)$, since s arbitrary.

Computer Science
UNIVERSITY OF TORONTO

RE identities

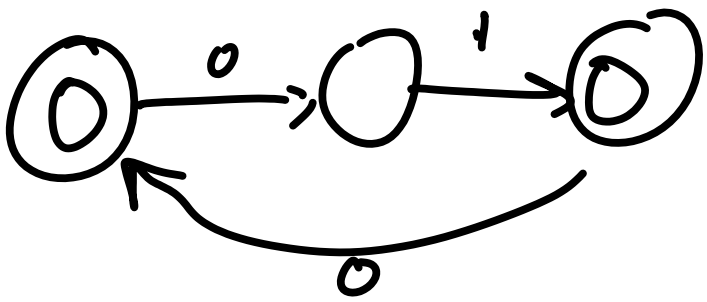
some of these follow from set properties...

others require some proof (see 7.2.5 example)

- communitativity of union: $R + S \equiv S + R$
- associativity of union: $(R + S) + T \equiv R + (S + T)$
- associativity of concatenation: $(RS)T \equiv R(ST)$
- left distributivity: $R(S + T) \equiv RS + RT$
- right distributivity: $(S + T)R \equiv SR + TR$
- identity for union: $R + \emptyset \equiv R$
- identity for concatenation: $R\varepsilon \equiv R \equiv \varepsilon R$
- annihilator for concatenation: $\emptyset R \equiv \emptyset \equiv R\emptyset$
- idempotence of Kleene star: $(R^*)^* \equiv R^*$

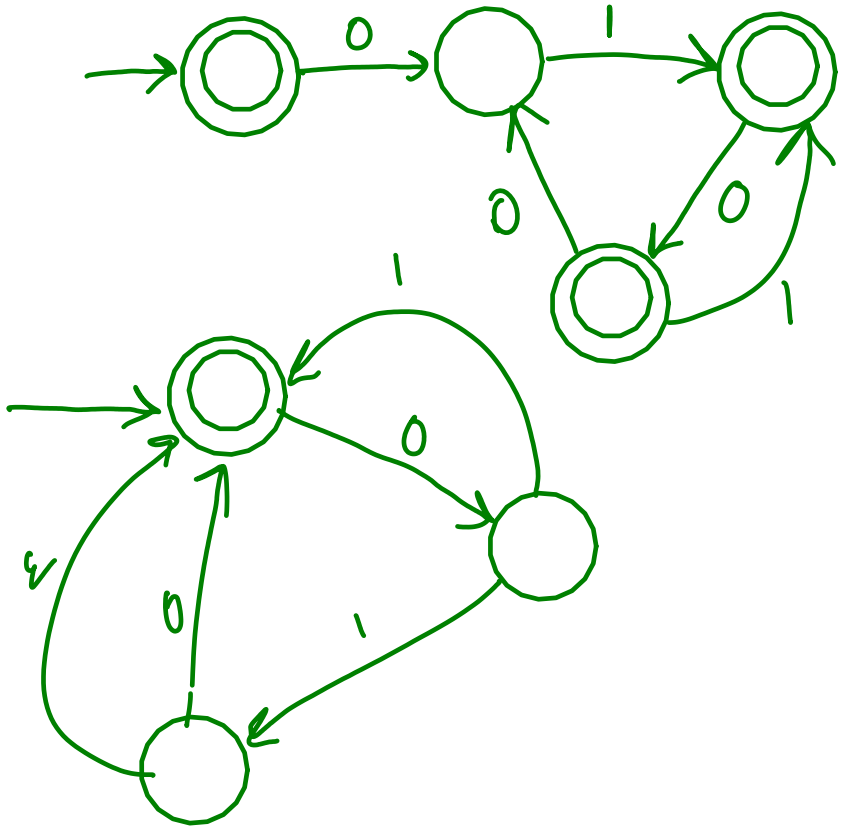
non-deterministic FSA (NFSA) example

FSA that accepts $L((010 + 01)^*$



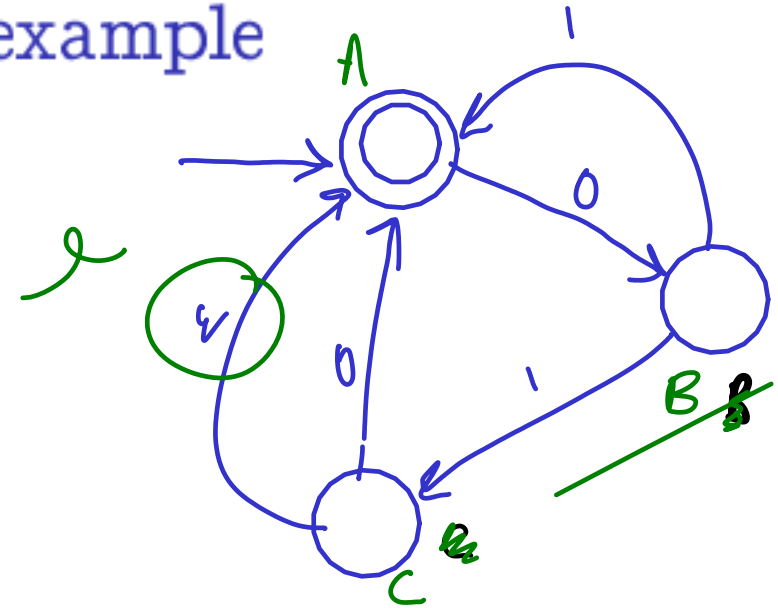
convenient!

ε
010
01
01001
01010

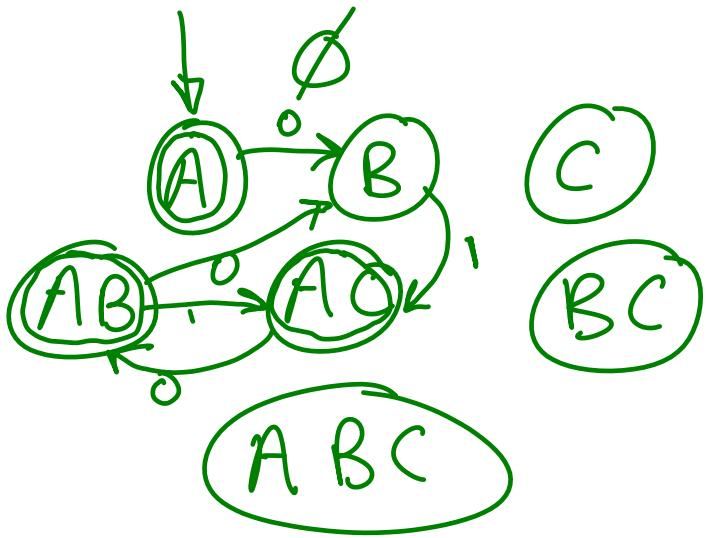


non-deterministic FSA (NFSA) example

FSA that accepts $L((010 + 01)^*$



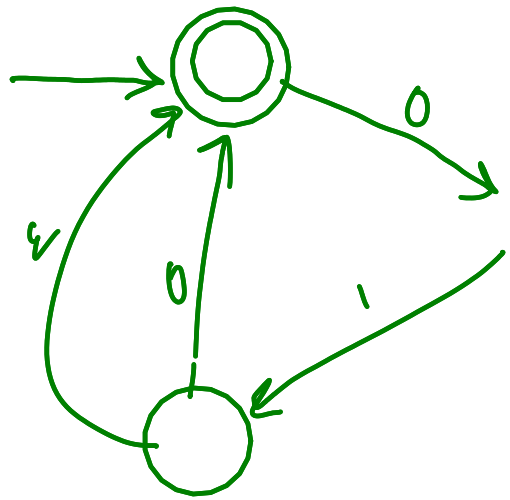
convenient!



non-deterministic FSA (NFSA) example

FSA that accepts $L((010 + 01)^*$

convenient!



NFSAs are real

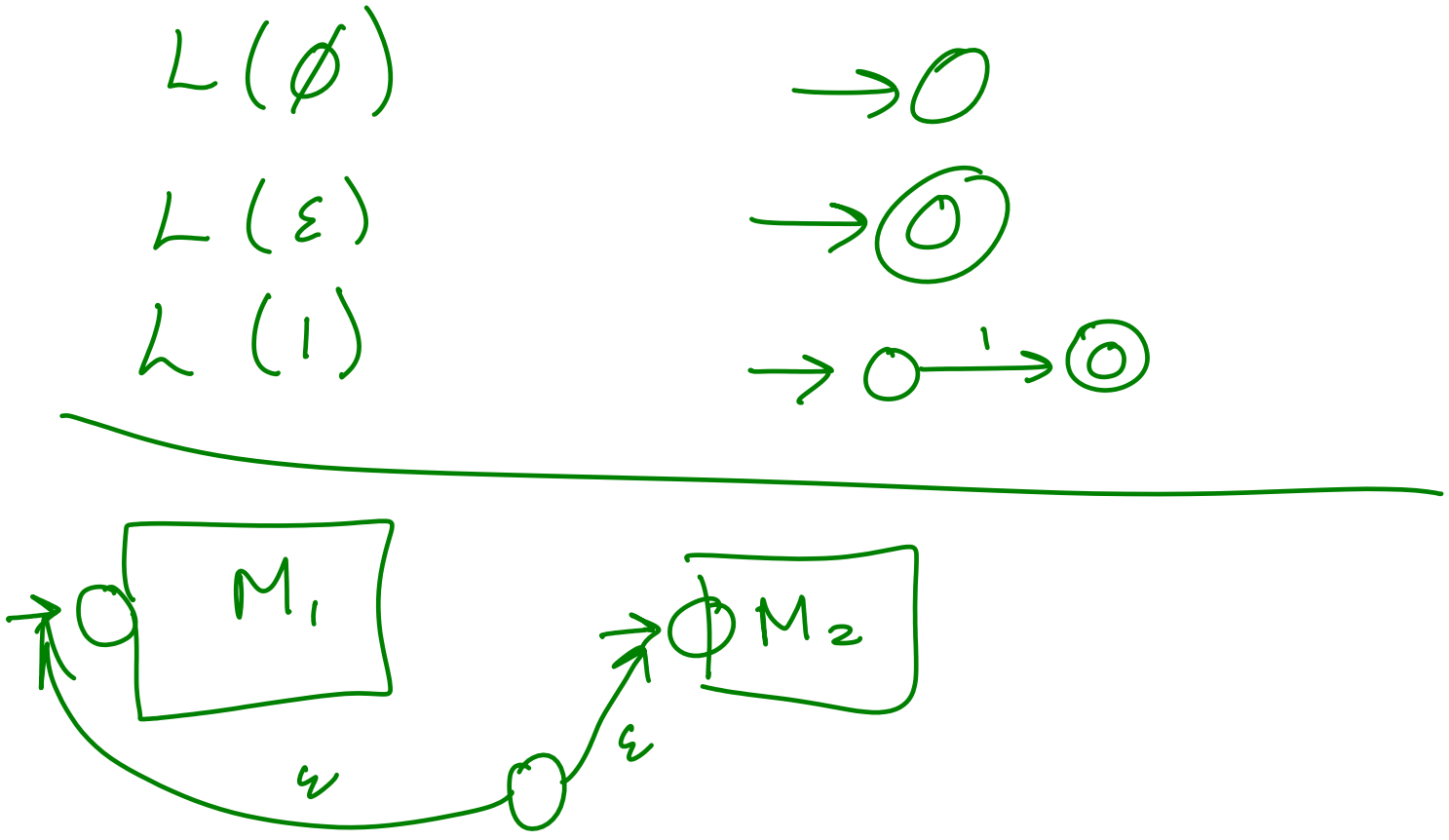
...you can always convert them to DFSA

29

Use subset construction, notes page 217

they're equivalent:

$L = L(M)$ for some DFSA $M \Leftrightarrow L = L(M')$ for some NFSA $M' \Leftrightarrow$
$L = R(R)$ for some regular expression R



notes