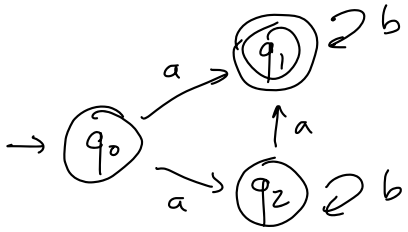
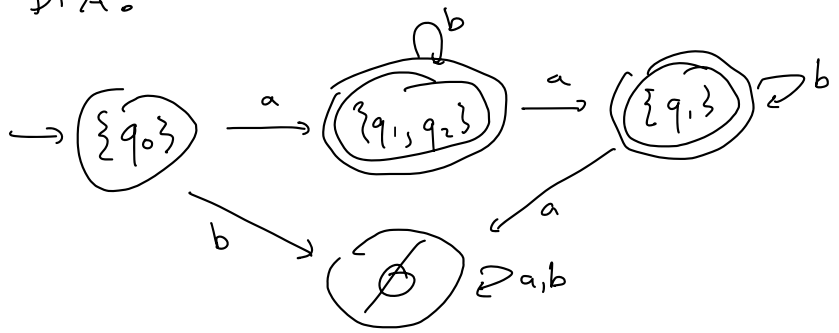
NFA:



Accepts ab^n or $ab^n ab^m$

Subset DFA:



This DFA simulates doing all paths in the NFA simultaneously. If one of the endpts of the path was accepted on the NFA, it gets accepted on the DFA.

The DFA simulates the NFA, so they have the same language.

So DFAs & NFAs compute the same languages: the regular languages.

Thm Given any language L,
$\exists$ a DFA M with $L = L(M)$

iff

$\exists$ a NFA N with $L = L(N)$

Thm Regular langs are closed under:

- complement ← swap accept/reject
- intersection } product of DFAs
- union
- difference
- concatenation ← 

Regular Expressions (regex)

Doesn't look like a machine – no states or transitions.

A reg. ex is a systematic way of rejecting or accepting strings.

Looks like: on $\Sigma = \{a, b\}$,

$$r = ((ab)^* + b^*ab)a$$

This defines a language. $L(r)$

$aba \in L(r)$ $\quad ba \notin L(r)$ etc.

A regex has: letters, with operations: $+$, $*$,
parentheses for grouping,
also ε or $\emptyset$

In a regex, letters or strings (incl ε) represent themselves.

So if $r = a$, then $L(r) = \{a\}$

if $r = aba$, then $L(r) = \{aba\}$

if $r = \varepsilon$, then $L(r) = \{\varepsilon\}$

if $r = \emptyset$, then $L(r) = \emptyset$

letters, ε, $\emptyset$ are called <u>atomics</u> regex.

Any regex is built from atomics using operations.

$+$: means "or" or $\cup$

$L(a+b) = \{a, b\}$

generally, $L(r_1 + r_2) = L(r_1) \cup L(r_2)$

*: "the Kleene star"

matches the preceding expression 0 or more times.

So: $L(a^*) = \{\varepsilon, a, a^2, a^3, \ldots\}$

$L((ab)^*) = \{\varepsilon, ab, (ab)^2, (ab)^3, \ldots\}$

$L(ab^*) = \{a, ab, ab^2, ab^3, \ldots\}$

$L((a+b)^*)$ includes $\varepsilon, a, a^2, a^3, a^4, \ldots$

or $\varepsilon, b, b^2, b^3, b^4, \ldots$

or $ababaaa$, bab

$(a+b)^*$ matches all strings on a, b.

$L((a+b)^*) = \{a, b\}^*$

$(a+b)^*$ is different from $a^* + b^*$

↑ matches everything.

↑ matches only a^n or b^n.

Concatenation ← writing one regex next to another.

$L(r_1 r_2)$ matches any string from $L(r_1)$ followed by any string from $L(r_2)$.

$L(a(a+b^2))$ matches anything with a, followed by a or b^2.

i.e. just aa or ab^2

$L((a+b^2)^*)$ any string made up of a's or b^2's.

$ab^2ab^2aab^2a$

or ab^4a^3 or a^{12} or b^8

$L(b(ab^* + a))$ matches strings like bab^n or ba.

$r = ((ab)^* + b^*ab)a$

matches $(ab)^n a$ or $b^n aba$

$\{(ab)^n a\} \cup \{b^n aba\}$

Any regex is a way of describing a set of strings. (sometimes regex is more convenient than writing in set theory)