Fundamentals of Computing

Coursework 1

MSc Data Science

Mark Rotchell
13181875

1

1.a

1.a.i

A	B	C	A ∨ B	A → C	C ∧ ¬B
0	0	0	0	1	0
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	1	0	0
1	1	1	1	1	0

The set is consistent for A=1, B=0 and C=1, as hightlighted in the above truth table

1.a.ii

A	B	C	¬A ∧ ¬B	¬C → A	¬C ∨ B
0	0	0	1	0	1
0	0	1	1	1	0
0	1	0	0	0	1
0	1	1	0	1	1
1	0	0	0	1	1
1	0	1	0	1	0
1	1	0	0	1	1
1	1	1	0	1	1

The set is not consistent as there exists no set of truth values for A, B, C such that all of the formulae in the set are true

1.b

A	B	C	$p_1 = \neg(A \wedge B)$	$p_2 = C \to A$	$p_3 = C \wedge B$	$(p_1 \wedge p_2 \wedge p_3) \to C$
0	0	0	1	1	0	1
0	0	1	1	0	0	1
0	1	0	1	1	0	1
0	1	1	1	0	1	1
1	0	0	1	1	0	1
1	0	1	1	1	0	1
1	1	0	0	1	0	1
1	1	1	0	1	1	1

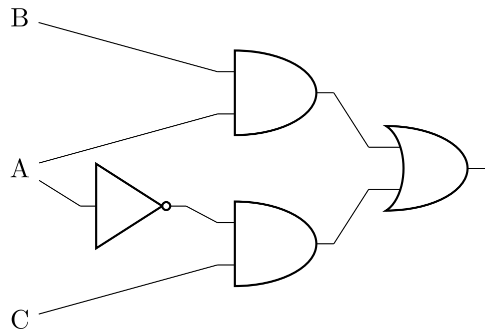
The argument is logically correct. There exist no situations in which all of the premises are true and the conclusion is false.

2

2.a

A	B	C	$A \to \neg B$	$C \to A$	$\neg((A \to \neg B) \wedge (C \to A))$
0	0	0	1	1	0
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	1	0	1
1	0	0	1	1	0
1	0	1	1	1	0
1	1	0	0	1	1
1	1	1	0	1	1

2.b



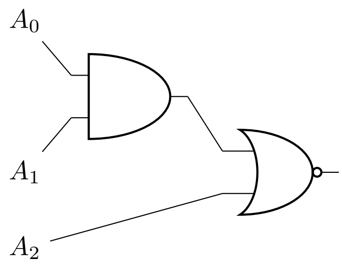
2.c

$$
\begin{aligned}
&\neg((A \to \neg B) \wedge (C \to A)) \\
\equiv&\neg((\neg A \vee \neg B) \wedge (C \to A)) \\
\equiv&\neg((\neg A \vee \neg B) \wedge (\neg C \vee A)) \\
\equiv&\neg((\neg B \vee \neg A) \wedge (\neg C \vee A)) \\
\equiv&\neg((B \to \neg A) \wedge (\neg C \vee A)) \\
\equiv&\neg(B \to \neg A) \vee \neg(\neg C \vee A) \\
\equiv&\neg(B \to \neg A) \vee (C \wedge \neg A) \\
\equiv&(B \to \neg A) \to (C \wedge \neg A)
\end{aligned}
$$

3

$$N = A_2 \cdot 2^2 + A_1 \cdot 2^1 + A_0 \cdot 2^0$$

A_2	A_1	A_0	$N < 3_{10}$	$A_1 \wedge A_0$	$A_2 \downarrow (A_1 \wedge A_0)$
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	0



4

The argument is not correct. With the following propositional variables:

A : 'Jones did not meet Smith last night'

B : 'Smith was the murderer'

C : 'Jones is lying'

D : 'The murder took place after midnight'

the propositions can be summarized as:

$p_1 = A \to (B \lor C)$

$p_2 = \neg B \to (A \land D)$

$p_3 = D \to (B \lor C)$

These propositions can all be true whilst the conclusion is false, as show by the below row from the truth table, therefore the argument is not correct

A	B	C	D	$p_1 = A \to (B \lor C)$	$p_2 = \neg B \to (A \land D)$	$p_3 = D \to (B \lor C)$
...					...	
1	0	1	1	1	1	1
...					...	

5

5.a

Initial digit is a `1` therefore number is negative. Finding absolute value via following steps:

Initial word	`1100 0001 1011 0000 0000 0000 0000 0000`
Invert digits	`0011 1110 0100 1111 1111 1111 1111 1111`
Add one	`1`
Result	`0011 1110 0101 0000 0000 0000 0000 0000`
	$= 2^{29} + 2^{28} + 2^{27} + 2^{26} + 2^{25} + 2^{22} + 2^{20} = 1,045,430,272_{10}$

Therefore the number is $-1,045,430,272_{10}$

5.b

`1100 0001 1011 0000 0000 0000 0000 0000`

$$= 2^{31} + 2^{30} + 2^{24} + 2^{23} + 2^{21} + 2^{20} = 3,249,537,024_{10}$$

5.c

$$1 \;\Big|\; \texttt{100 0001 1} \;\Big|\; \texttt{011 0000 0000 0000 0000 0000}$$

Sign bit is `1`, so $S = 1$.
Exponent bits are `100 0001 1` $= 2^7 + 2^1 + 2^0 = 131$, and the bias is 127, so $E = 131 - 127 = 4$.
Fraction bits are `011 000` $\ldots$, so $F = 1 + 2^{-2} + 2^{-3} = 1\frac{3}{8}$
So number is $(-1)^S \cdot F \cdot 2^E = -1 \cdot 1\frac{3}{8} \cdot 2^4 = -22$

6

6.a

First find binary representation of $|-107| = 107$

$$\begin{aligned}
107/2 &= 53 \text{ remainder } 1\\
53/2 &= 26 \text{ remainder } 1\\
26/2 &= 13 \text{ remainder } 0\\
13/2 &= 6 \text{ remainder } 1\\
6/2 &= 3 \text{ remainder } 0\\
3/2 &= 1 \text{ remainder } 1\\
1/2 &= 0 \text{ remainder } 1
\end{aligned}$$

reading updwards $107_{10} = \texttt{1101011}_2$, then convert to negative in two's complement via

Initial word	0000 0000 0000 0000 0000 0000 0110 1011
Invert digits	1111 1111 1111 1111 1111 1111 1001 0100
Add one	1
Result	1111 1111 1111 1111 1111 1111 1001 0101

6.b

$$-107_{10} = -110\ 1011_2 = -1.1010\ 11_2 \times 2^6$$

So, sign bit is `1`
Exponent is $6_{10} + 127_{10} = 133_{10} = \texttt{1000 0101}_2$
Fraction is 101011
And full word is

1100 0010 1101 0110 0000 0000 0000 0000

6.c

$$-14.375 = -\left(8 + 4 + 2 + \frac{1}{4} + \frac{1}{8}\right) = -1110.011_2 = -1.1100\ 11_2 + 2^3$$

So, sign bit is `1`
Exponent is $3_{10} + 127_{10} = 130_{10} = \texttt{1000 0010}_2$
Fraction is 1100 11
And full word is

1100 0001 0110 0110 0000 0000 0000 0000

7

7.a

$$f(x) = x + 1$$

7.b

$$f(x) = \begin{cases} \dfrac{x}{2} & \text{if } x \text{ is even} \\ \\ \dfrac{x-1}{2} & \text{if } x \text{ is odd} \end{cases}$$

7.c

$$f(x) = \begin{cases} x + 2 & \text{if 3 divides } x \\ x - 1 & \text{if 3 does not divide } x \end{cases}$$

7.d

$$f(x) = 0$$

8

8.a

Note that $f(2) = (2+1)/(2-2) = 3/0$ is undefined, so f is not a function over the whole real domain as stated in the question. As it is not a function, it cannot be an injective nor surjective function, so $\underline{f \text{ is not injective (one-to-one)}}$ and $\underline{f \text{ is not surjective (onto)}}$.

8.b

Injectivity

$$f(0,1) = 0 = f(1,0)$$

however

$$(0,1) \neq (1,0)$$

$$\therefore \neg\left(\forall (m,n),(p,q) \in \mathbb{Z} \times \mathbb{Z} : f(m,n) = f(p,q) \implies (m,n) = (p,q)\right)$$

therefore $\underline{f \text{ is not injective (one-to-one)}}$

Surjectivity Suppose $n = 1$ then $f(m,n) = f(m,1) = m + 1 - 1 = m$. So,

$$\forall y \in \mathbb{Z} : f(y,1) = y$$

$$\therefore \ \forall y \in \mathbb{Z},\ \exists (m,n) \in \mathbb{Z} \times \mathbb{Z},\ f(m,n) = y$$

therefore $\underline{f \text{ is surjective (onto)}}$

8.c

Injectivity

$$f(0,-1) = 0 = f(-1,0)$$

however

$$(0,-1) \neq (-1,0)$$

$$\therefore \neg\left(\forall (m,n),(p,q) \in \mathbb{Z} \times \mathbb{Z} : f(m,n) = f(p,q) \implies (m,n) = (p,q)\right)$$

therefore $\underline{f \text{ is not injective (one-to-one)}}$

Surjectivity

$$f(\max(0, y+1), \min(0, y+1)) = y$$

$$\forall y \in \mathbb{Z},\ \max(0, y+1) \in \mathbb{Z},\ \min(0, y+1) \in \mathbb{Z}$$

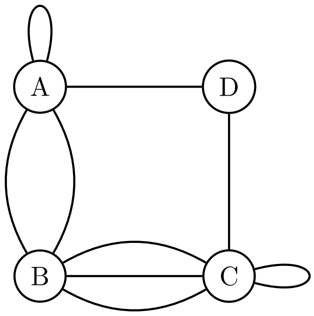
$$\therefore\ \forall y \in \mathbb{Z},\ \exists (m, n) \in \mathbb{Z} \times \mathbb{Z},\ f(m, n) = y$$

therefore f is surjective (onto)

9

9.a

	A	B	C	D
A	1	2	0	1
B	2	0	3	0
C	0	3	1	1
D	1	0	1	0



The graph is not simple as there are vertices with more than one edge between them.

9.b

Let the left hand graph be G such that

$$G = (V_G, E_G) = (\{1, 2, 3, 4, 5\}, \{1, 2\}, \{1, 3\}, \{1, 5\}, \{2, 3\}, \{2, 4\}, \{4, 5\})$$

and the right graph be H such that

$$H = (V_H, E_H) = (\{a, c, b, d, e\}, \{a, c\}, \{a, b\}, \{a, e\}, \{c, b\}, \{c, d\}, \{d, e\})$$

Then let the function $f : V_G \to V_H$ be the vertex bijection given by the following set of pairs of the form $(v, f(v))$:

$$\{(1, a), (2, c), (3, b), (4, d), (5, e)\}$$

Applying the function f to each vertex of each edge $\{v_1, v_2\}$ in E_G gives a set of subsets, $\{f(v_1), f(v_2)\} \subset V_H$ which is exactly E_H, i.e. all the edges of H.

$\{v_1, v_2\}$	$\{f(v_1), f(v_2)\}$
{1,2}	{a,c}
{1,3}	{a,b}
{1,5}	{a,e}
{2,3}	{c,b}
{2,4}	{c,d}
{4,5}	{d,e}

All the edges from the two graphs are in above table. No set of vertices which isn't an edge is in the table. If two vertices are adjacent in G their f-mapped vertices in H are also adjacent. If two vertices are not adjacent in G their f-mapped vertices in H are also not adjacent. Formally

$$\{v_1, v_2\} \in E_G \iff \{f(v_1), f(v_2)\} \in E_H$$

therefore f is an isomorphism and G and H are isomorphic.

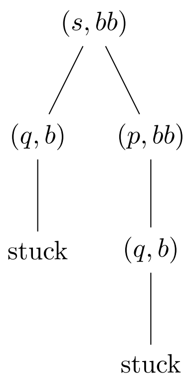
9.c

The two graphs have a different number of edges, which is an invariant, so the two graphs cannot be isomorphic.

10

10.a

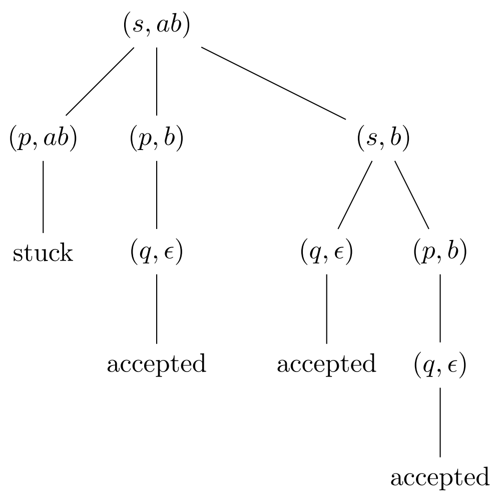
Computations for bb:



- $(s, bb), (q, b) \rightsquigarrow$ stuck
- $(s, bb), (p, bb), (q, b) \rightsquigarrow$ stuck

word bb is not accepted

Computations for ab:

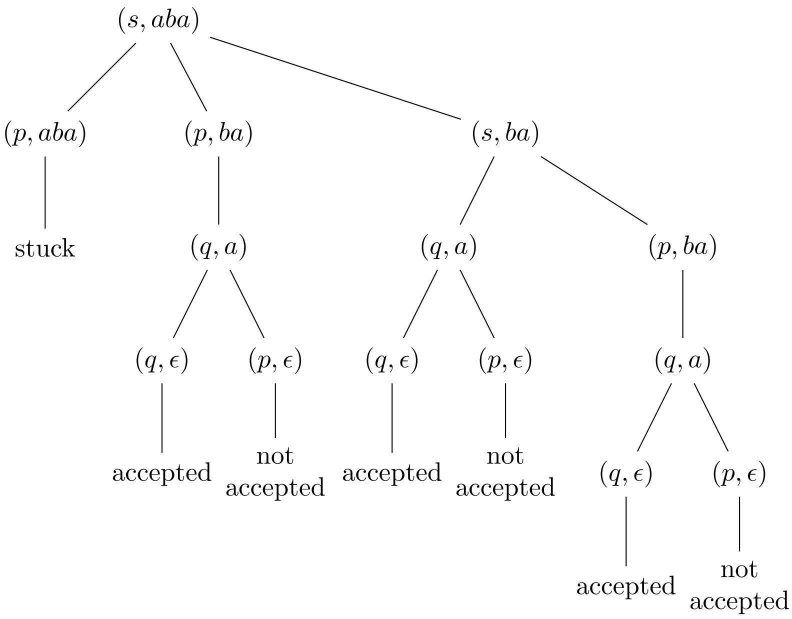


- $(s, ab), (p, ab) \rightsquigarrow$ stuck
- $(s, ab), (p, b), (q, \epsilon) \rightsquigarrow$ accepted
- $(s, ab), (s, b), (q, \epsilon) \rightsquigarrow$ accepted

- $(s, ab), (s, b), (p, b), (q, \epsilon) \rightsquigarrow$ accepted

word ab is accepted

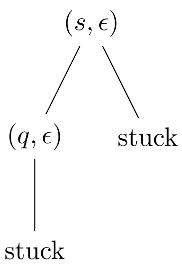
Computations for aba:



- $(s, aba), (p, aba) \rightsquigarrow$ stuck
- $(s, aba), (p, ba), (q, a), (q, \epsilon) \rightsquigarrow$ accepted
- $(s, aba), (p, ba), (q, a), (p, \epsilon) \rightsquigarrow$ not accepted
- $(s, aba), (s, ba), (q, a), (q, \epsilon) \rightsquigarrow$ accepted
- $(s, aba), (s, ba), (q, a), (p, \epsilon) \rightsquigarrow$ not accepted
- $(s, aba), (s, ba), (p, ba), (q, a), (q, \epsilon) \rightsquigarrow$ accepted
- $(s, aba), (s, ba), (p, ba), (q, a), (p, \epsilon) \rightsquigarrow$ not accepted

word aba is accepted

Computations for ϵ:

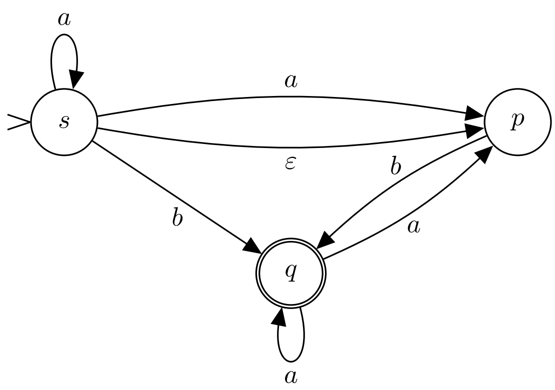


- $(s, \epsilon) \rightsquigarrow$ not accepted

- $(s, \epsilon), (p, \epsilon) \rightsquigarrow$ not accepted

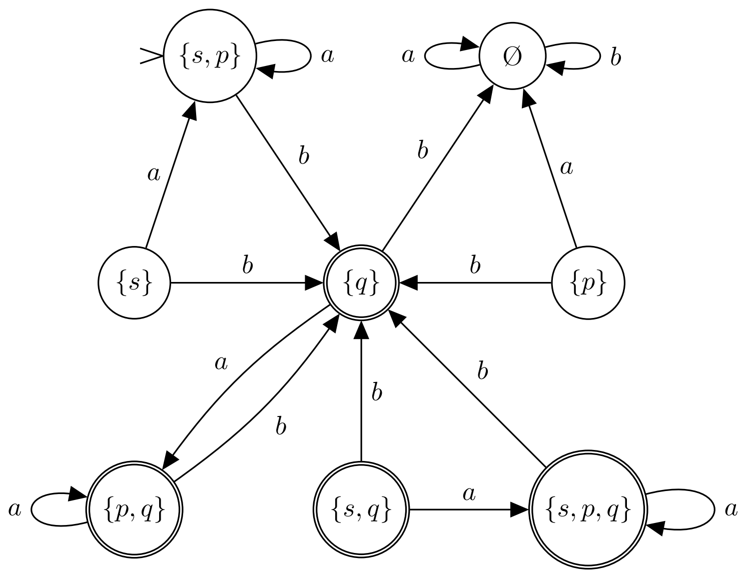
word ϵ is not accepted

10.b

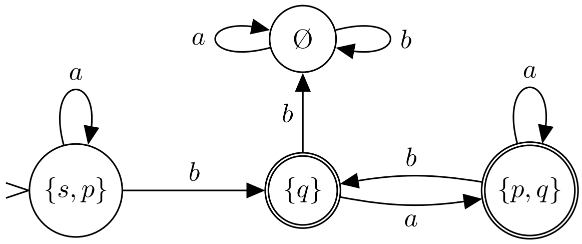


subset of states	reachable by a followed by one or more ε	reachable by b followed by one or more ε
$\{s\}$	$\{s, p\}$	$\{q\}$
$\{p\}$	$\emptyset$	$\{q\}$
$\{q\}$	$\{p, q\}$	$\emptyset$
$\{s, p\}$	$\{s, p\}$	$\{q\}$
$\{s, q\}$	$\{s, p, q\}$	$\{q\}$
$\{p, q\}$	$\{p, q\}$	$\{q\}$
$\{s, p, q\}$	$\{s, p, q\}$	$\{q\}$
$\emptyset$	$\emptyset$	$\emptyset$

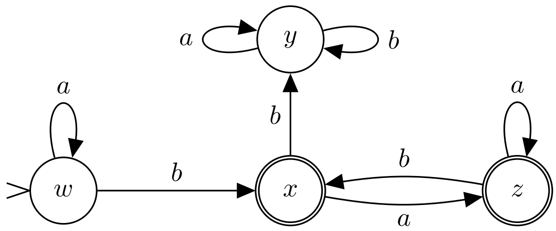
The DFA starting state is the ε-closure of the NFA starting state, which is $\{s, p\}$. All subsets with q are accepting states. Full subset construction:



then removing the unreachable states:



then re-labelling the states



we have a deterministic finite automaton $A = (Q, \Sigma, \delta, q_0, F)$ with

- a set of states $Q = \{w, x, y, z\}$
- a set of symbols $\Sigma = \{a, b\}$
- a transition function $f : Q \times \Sigma \to Q$ given by the below table

	symbol	
state	a	b
w	w	x
x	z	y
y	y	y
z	z	x

- an initial state $q_0 = w$
- a set of accepting states $F = \{x, z\}$

10.c

The automaton accepts words comprised of a and b that contain at least one b, but no consecutive bs.

10.d

`a*b(aa*b)*a*`

10.e

Let $G = (V, \Sigma, R, s)$ be a context free grammar with

- a set of variables $V = \{S, X, Y\}$
- a set of terminals $\Sigma = \{a, b\}$
- a set of production rules $R = \{S \to XbYX,\ X \to \varepsilon,\ X \to Xa,\ Y \to \varepsilon,\ Y \to Ya, T \to Yab\}$
- a start variable $s = S$

11

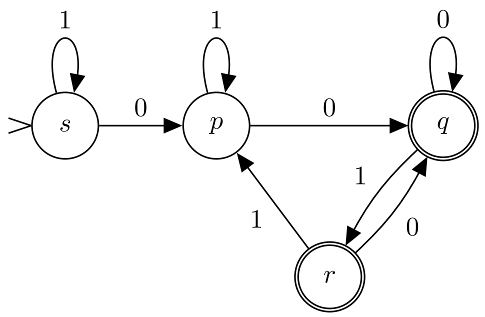
Let $A = (Q, \Sigma, \delta, q_0, F)$ be a deterministic finite automaton with

- a set of states $Q = \{s, p, q, r\}$
- a set of symbols $\Sigma = \{1, 0\}$
- a transition function $f : Q \times \Sigma \to Q$ given by the below table

	symbol	
state	0	1
s	p	s
p	q	p
q	q	r
r	q	p

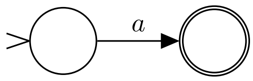
- an initial state $q_0 = s$
- a set of accepting states $F = \{q, r\}$

We can visualise A with the following state diagram

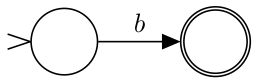


This can be represented by the following regular expression $\mathtt{1}^*\mathtt{0}(\mathtt{0} \cup \mathtt{1})^*(\mathtt{0} \cup \mathtt{01})$

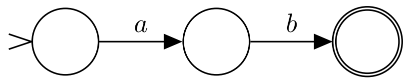
12



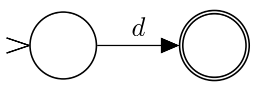
Automaton accepting $L[a]$



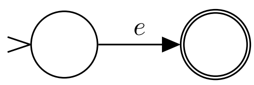
Automaton accepting $L[b]$



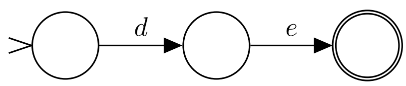
Automaton accepting $L[ab]$



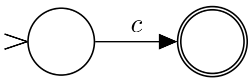
Automaton accepting $L[d]$



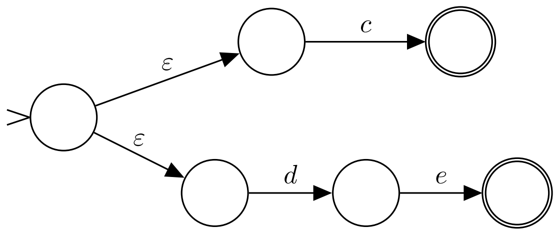
Automaton accepting $L[e]$



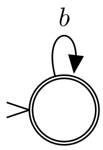
Automaton accepting $L[de]$



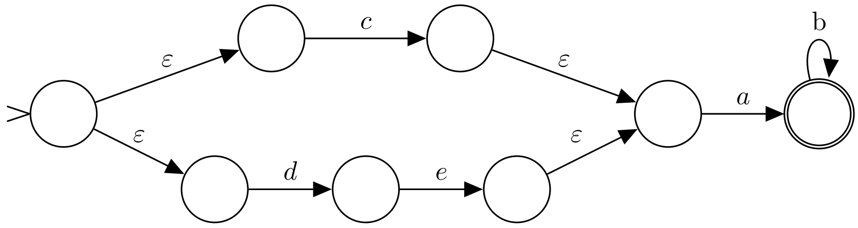
Automaton accepting $L[c]$



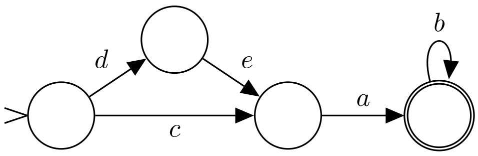
Automaton accepting $L[c \cup de]$



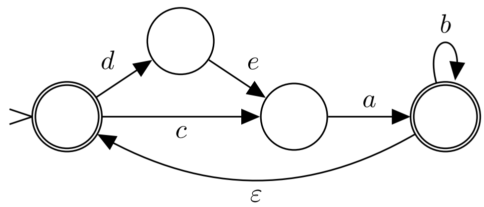
Automaton accepting $L[b^*]$



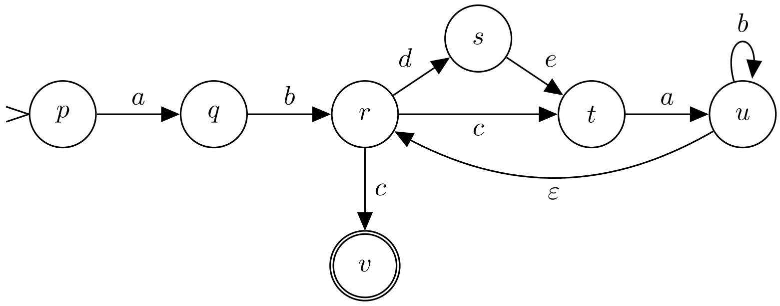
Automaton accepting $L[(c \cup de)ab^*]$



Simplified Automaton accepting $L[(c \cup de)ab^*]$



Automaton accepting $L[\big((c \cup de)ab^*\big)^*]$



Automaton accepting $L[ab\big(c \cup de)ab^*\big)^*c]$

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a nondeterministic finite automaton with ε moves, comprising

- a set of states $Q = \{p, q, r, s, t, u, v\}$
- a set of symbols $\Sigma = \{a, b, c, d, e\}$
- a transition function $f : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathsf{Pow}(Q)$ given by the below table

	symbol					
state	a	b	c	d	e	ε
p	$\{q\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
q	$\varnothing$	$\{r\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
r	$\varnothing$	$\varnothing$	$\{t, v\}$	$\{s\}$	$\varnothing$	$\varnothing$
s	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{t\}$	$\varnothing$
t	$\{u\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
u	$\varnothing$	$\{u\}$	$\varnothing$	$\varnothing$	$\varnothing$	$\{r\}$
v	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

- an initial state $q_0 = p$
- a set of accepting states $F = \{v\}$

Then A accepts the language $L[ab\big(c \cup de)ab^*\big)^* c]$.

13

13.a

`aaa*ba*`

13.b

$L = \{a^n b^m a^n | n \geq 0, m \geq 3\}$

L is not regular. Consider words of the form $a^n bbba^n$ which are all in L and can be of any length greater than 3 – no substring can be removed from these words whilst they remain in L: if the substring contains any b, and that substring is removed, there will be fewer than 3 b; if the substring contains only a, and that substring is removed, there will different numbers of a on each side. As there is no substring that can be removed, there is no substring that can be pumped. Therefore, for any length greater than 3 there is a word in L which does not satisfy the pumping property, therefore it is not regular.

14

Let $G = (V, \Sigma, R, s)$ be a context free grammar with

- a set of variables $V = \{S, X, Y\}$
- a set of terminals $\Sigma = \{a, b, c\}$
- a set of production rules $R = \{S \to XY,\ X \to \varepsilon,\ X \to aXb,\ Y \to \varepsilon,\ Y \to bYc\}$
- a start variable $s = S$

This would be accepted by a push-down automaton, $P = (Q, \Sigma, \Gamma, \delta, s_0, F)$ with

- a set of states, $Q = \{p, q, r, s, t, u\}$
- an input alphabet, $\Sigma = \{a, b, c\}$

- a stack alphabet, $\Gamma = \{\perp, a, b\}$
- a transition relation, δ, consisting of the following instructions of the form $((q_0, \sigma, \gamma_0), (q_1, \gamma_1))$
  - $((p, \varepsilon, \varepsilon), (q, \perp))$
  - $((q, a, \varepsilon), (q, a))$
  - $((q, \varepsilon, \varepsilon), (r, \varepsilon))$
  - $((r, b, a), (r, \varepsilon))$
  - $((r, \varepsilon, \perp), (s, \perp))$
  - $((s, b, \varepsilon), (s, b))$
  - $((s, \varepsilon, \varepsilon), (t, \varepsilon))$
  - $((t, c, b), (t, \varepsilon))$
  - $((t, \varepsilon, \perp), (u, \varepsilon))$

  where

  - $q_0 \in Q$ is the starting state of the transition
  - $\sigma \in (\Sigma \cup \{\varepsilon\})$ is the input symbol read during the transition
  - $\gamma_0 \in (\Gamma \cup \{\varepsilon\})$ is the symbol popped from the stack during the transition
  - $q_1 \in Q$ is the ending state of the transition
  - $\gamma_1 \in (\Gamma \cup \{\varepsilon\})$ is the symbol pushed onto the stack during the transition
- an initial state, $s_0 = p$
- a set of accepting states $F = \{u\}$

P can be represented by the following state diagram: