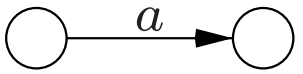
Chapter 1

Review of Formal Languages and Automata Theory

	grammar	**automaton**
3	regular	
	right-linear $A \to aB$	finite state 
2	context-free	
	$A \to \alpha$	pushdown (+lifo stack)
1	context-sensitive	
	$(\beta_\ell, A, \beta_r) \to \alpha$ $\alpha \to \beta \quad \lvert\beta\rvert \geq \lvert\alpha\rvert$ monotone	linear bounded
0	recursively enumerable	
	$\alpha \to \beta$	turing machine

1.1 Sets
1.2 Symbols, strings, and languages
1.3 Regular expressions and regular languages
1.4 Finite automata
1.5 Context-free grammars and languages
1.6 Turing machines
1.7 Unsolvability
1.8 Complexity theory

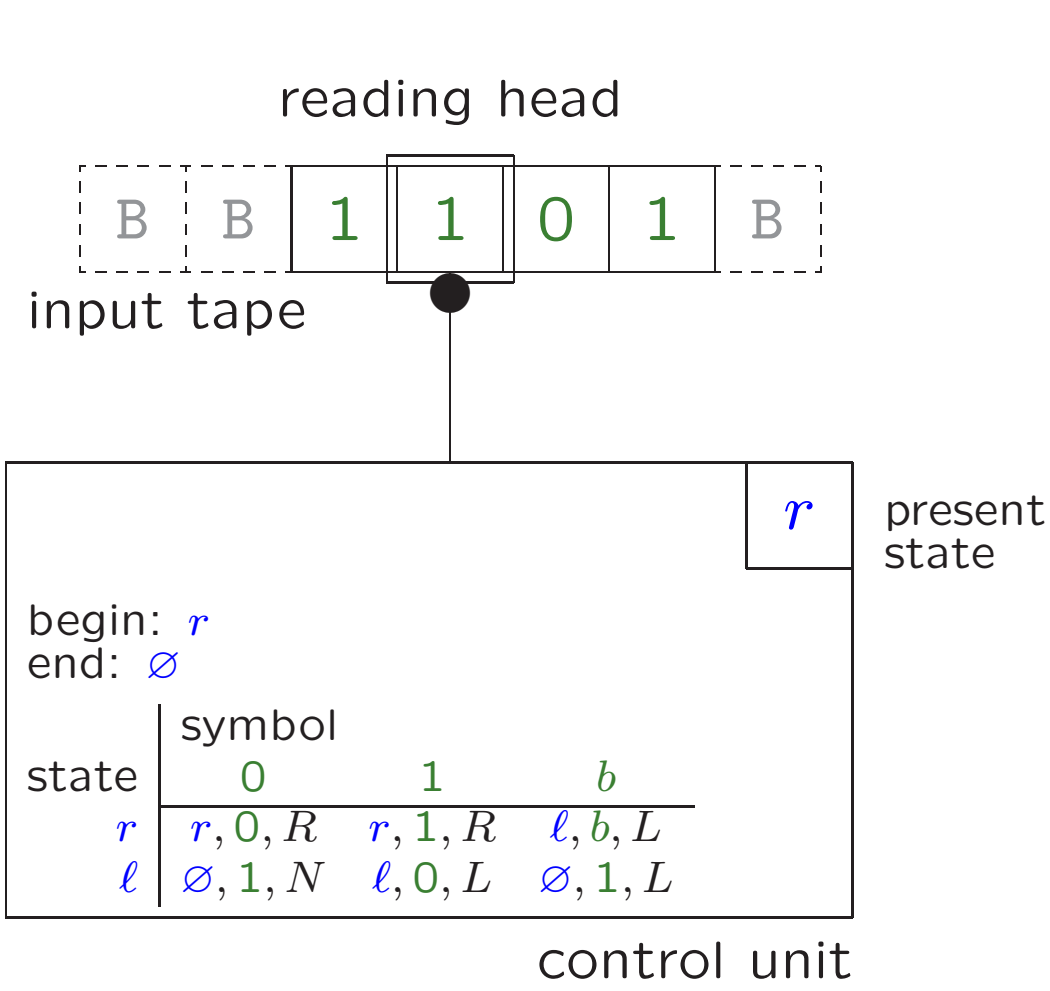
1.4 Finite automata

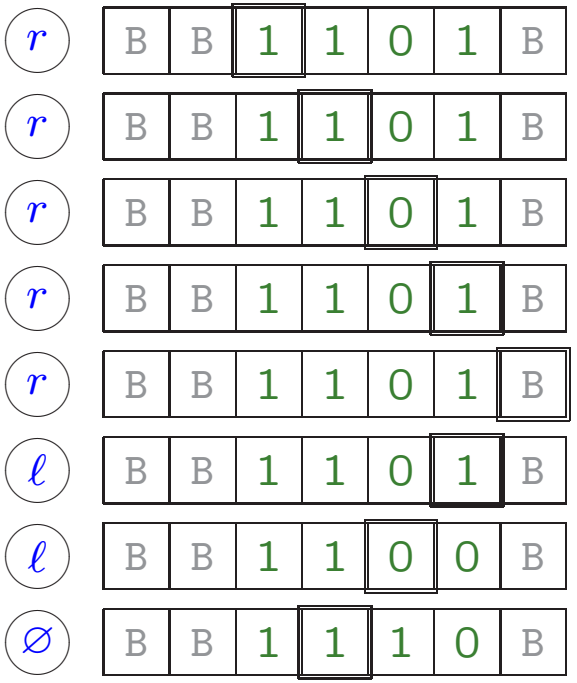
MOVED to Chapter 3

1.5 Context-free grammars and languages

MOVED to Chapter 4

1.6 Turing machines


reading head
B B 1 1 0 1 B
input tape
r
present state
begin: r
end: ∅
symbol
state 0 1 b
r r, 0, R r, 1, R ℓ, b, L
ℓ ∅, 1, N ℓ, 0, L ∅, 1, L
control unit


r B B 1 1 0 1 B
r B B 1 1 0 1 B
r B B 1 1 0 1 B
r B B 1 1 0 1 B
r B B 1 1 0 1 B
ℓ B B 1 1 0 1 B
ℓ B B 1 1 0 0 B
∅ B B 1 1 1 0 B

$\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, h)$

Q	states
$q_0 \in Q$	initial state
$h \in Q$	halting state
Γ	tape alphabet
$\Sigma \subseteq \Gamma$	input alphabet $\quad \mathtt{B} \in \Gamma - \Sigma$

transition function (partial)

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

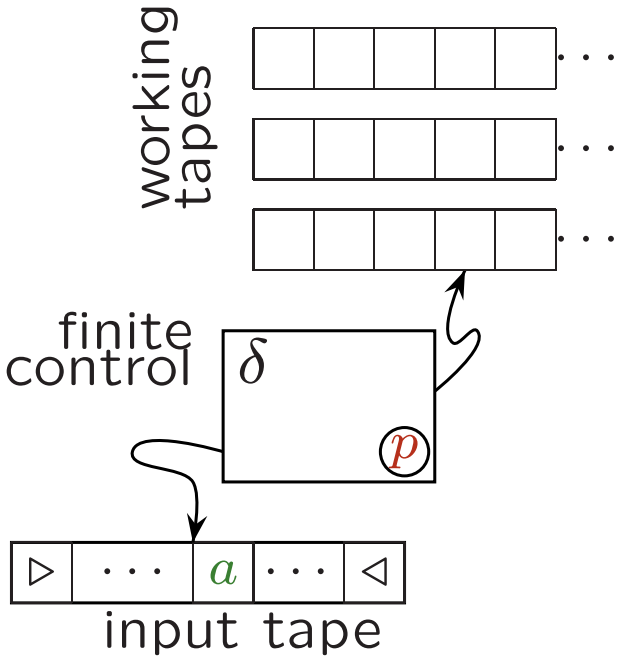
$$\text{(nondet) } \delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R, S\}$$

wqx configuration $w, x \in \Gamma^*,\ q \in Q$

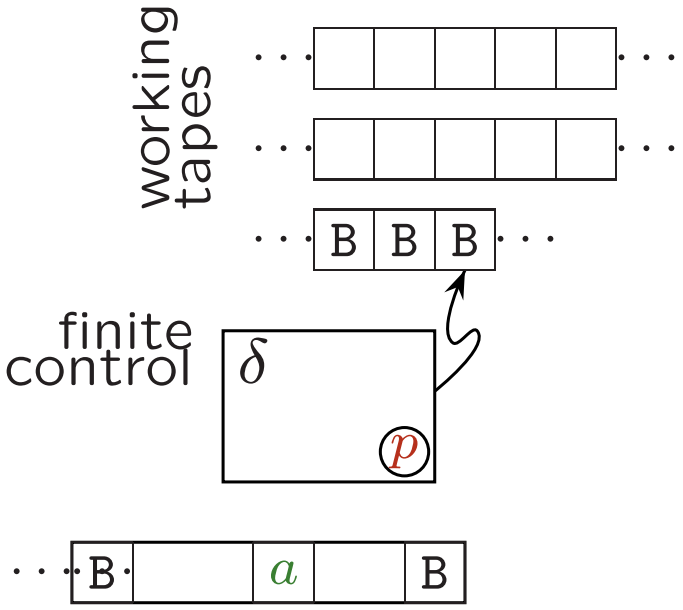
$\alpha Z p X \beta \vdash \alpha q Z Y \beta$	if	$\delta(p, X) = (q, Y, L)$
$\alpha p X \beta \vdash \alpha Y q \beta$	if	$\delta(p, X) = (q, Y, R)$
$\alpha p X \beta \vdash \alpha q Y \beta$	if	$\delta(p, X) = (q, Y, S)$

$L(\mathcal{M}) =$

$\{\ x \in \Sigma^* \mid q_0 \mathtt{B} x \vdash^* \alpha h \beta \text{ for some } \alpha, \beta \in \Gamma^*\ \}$



DSPACE(f) NSPACE(f)
space complexity
online Turing machine
multiple working tapes
single sided



DTIME(f) NTIME(f)
time complexity
input on tape
multiple working tapes
double sided

- blocks: no move defined		*'no'*
- move off tape (one-sided)		*'no'*
- infinite computation 'loop' cannot tell)		*'no'* (but we
- halt in state h		*'yes'*

RE	recursively enumerable	*enumerate*
REC	recursive — TM always stops	*decide*

REC closed complementation, RE is not

equivalent variants:
multiple tapes
one sided, two sided
two-dimensional tape
nondeterminism

> “ It is possible to invent a single machine which can be used to compute any computable sequence. If this machine $\mathcal{U}$ is supplied with a tape on the beginning of which is written the S.D. [=description] of some computing machine $\mathcal{M}$, then $\mathcal{U}$ will compute the same sequence as $\mathcal{M}$. ”

A.M. Turing, *On Computable Numbers, with an Application to the Entscheidungsproblem.* Proc. London Math. Soc. Ser. 2 42, 230-265, 1937. doi:10.1112/plms/s2-42.1.230

universal TM $\mathcal{M}_U$:
on input $e(T)e(x)$, $\mathcal{M}_U$ simulates T on input x and halts if and only if T halts on x.

$\Gamma_U = \{\mathsf{B} = a_0, a_1, a_2, \dots\}$
$Q_U = \{q_0, q_1, q_2, \dots\}$
$e(a_i) = 0^{i+1}$ $e(q_i) = 0^{i+1}$

alphabet $\Sigma = \{b_1, \dots, b_r\}$
$e(\Sigma) = 111\, e(b_1)\, 1\, e(b_2)\, 1 \cdots 1\, e(b_r)\, 111$

$e(L) = 0,\ e(R) = 00,\ e(S) = 000$

instruction m: $\delta(q, a) = (p, b, D)$
$e(m) = 11\, e(q)\, 1\, e(a)\, 1\, e(p)\, 1\, e(b)\, 1\, e(D)\, 11$

TM $\mathcal{M}$ with instructions $m_1, \dots, m_t$
$e(\mathcal{M}) =$
$11111\, e(q_0)\, 1\, e(h)\, 1\, e(\Sigma)\, 1\, e(\Gamma)\, 1\, e(m_1)\, 1 \cdots 1\, e(m_t)\, 11111$

1.7 Unsolvability

problem $\equiv$ language

$L_P = \{$ code$(x) \mid P(x)$ holds $\}$ 'yes' instances

P solvable $\equiv$ L_P recursive

halting problem:
given Turing machine T and input w, does T halt on w?

unsolvable (undecidable, uncomputable)

halting language
$\{ e(T)e(w) \mid T$ halts on $w \}$
in RE $-$ REC

> “ We can show further that *there can be no machine* $\mathcal{E}$ *which, when supplied with the S.D of an arbitrary machine* $\mathcal{M}$*, will determine whether* $\mathcal{M}$ *ever prints a given symbol (*0 *say).* ”

1.8 Complexity theory

Walter J. Savitch Deterministic simulation of non-deterministic turing machines (Detailed Abstract)

STOC '69 Proceedings of the first annual ACM symposium on Theory of computing 247 - 248 (1969)

10.1145/800169.805439

time complexity $T(n)$

P vs. NP

Karp reduction $L_1 \leq L_2$

$x \in L_1$ iff $f(x) \in L_2$

f computable in polynomial time

NP complete:

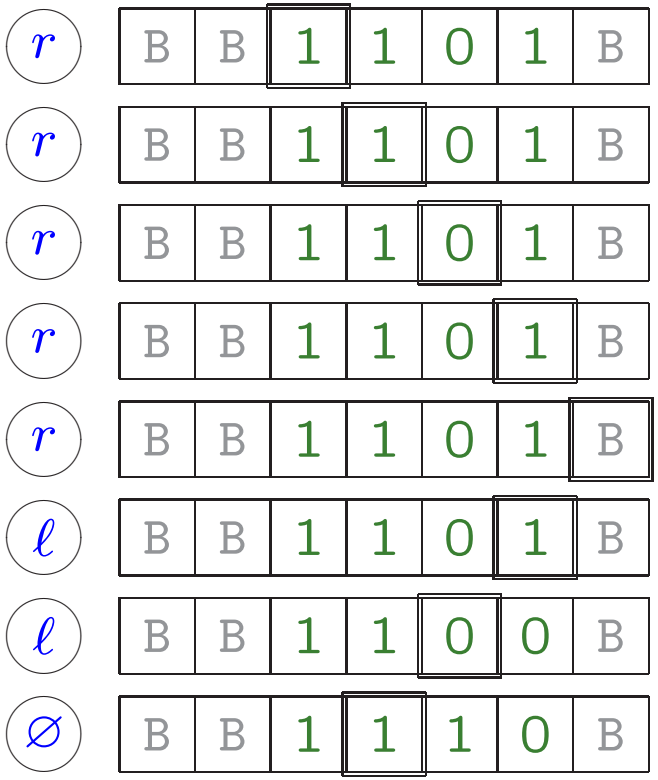
- $L \in$ NP
- $L' \leq L$ for every $L' \in$ NP

Savitch

NSPACE$(f(n)) \subseteq$ DSPACE$(f(n)^2)$

$f(n) \geq \log n$

PSPACE = NPSPACE



at step k

T_{iak} tape cell i contains symbol a

H_{ik} read/write head is at tape cell i

Q_{qk} M is in state q

computation M on x $\Longleftrightarrow$ φ satisfiable

▷

Cook, Stephen (1971). "The complexity of theorem proving procedures". Proceedings of the Third Annual ACM Symposium on Theory of Computing. pp. 151–158.

Levin, Leonid (1973). "Universal search problems (in Russian, Universal'nye perebornye zadachi)". Problems of Information Transmission (Russian: Problemy Peredachi Informatsii) 9 (3): 265–266. (Russian), translated into English by Trakhtenbrot, B. A. (1984). "A survey of Russian approaches to perebor (brute-force searches) algorithms". Annals of the History of Computing 6 (4): 384–400. doi:10.1109/MAHC.1984.10036

wikipedia:Cook-Levin Theorem

◁

$(Q_1 x_1)(Q_2 x_2) \cdots (Q_n x_n)\, \phi(x_1, x_2, \ldots, x_n)$

$A = Q_2 x_2 \cdots Q_n x_n\, \phi(0, x_2, \ldots, x_n)$

$B = Q_2 x_2 \cdots Q_n x_n\, \phi(1, x_2, \ldots, x_n)$

use recursion

stack depth max n

closed formula

SAT is NP-hard

variables

T_{iak} tape cell i contains symbol a at step k

H_{ik} read/write head is at tape cell i at step k

Q_{qk} M is in state q at step k

conjunction of

$T_{ix[i]0}$ initial tape $x[i] = x_i$ or $x[i] = \mathtt{B}$

$Q_{q_0 0}$ initial state

H_{00} initial position head

$T_{iak} \to \neg T_{ibk}$ one symbol per tape cell $a \neq b$

$Q_{pk} \to \neg Q_{qk}$ one state at a time $p \neq q$

$H_{ik} \to \neg H_{jk}$ one head position at a time $i \neq j$

$T_{iak} \wedge T_{ib(k+1)} \to H_{ik}$ tape changed only if written $a \neq b$

$H_{ik} \wedge Q_{pk} \wedge T_{iak} \to \bigvee_{(p,a,q,b,d)\in\delta} H_{(i+d)(k+1)} \wedge Q_{q(k+1)} \wedge T_{ib(k+1)}$
possible transitions

$Q_{h\,p(n)}$ finish in accepting state h

CNF conjunctive normal form

conjunction $\bigwedge$ clauses (disjunction $\bigvee$ literals)
$(x_1 \vee x_4) \wedge (x_2 \vee x_3 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_4 \vee x_5)$

$a \to \neg b \iff \neg a \vee \neg b$

$a \wedge b \to c \iff \neg a \vee \neg b \vee c$

and finally
$a \wedge b \wedge c \to \bigvee_{i \in I}(d_i \wedge e_i \wedge f_i) \iff *$
$(\neg a \vee \neg b \vee \neg c \vee \bigvee_{i \in I} z_i) \wedge \bigwedge_{i \in I}(\neg z_i \vee d_i) \wedge \bigwedge_{i \in I}(\neg z_i \vee e_i) \wedge \bigwedge_{i \in I}(\neg z_i \vee f_i)$

3SAT is NP-complete

$a \vee b \vee c \vee d \vee e \iff * \ (a \vee b \vee x_1) \wedge (\neg x_1 \vee c \vee x_2) \wedge (\neg x_2 \vee d \vee e)$

3SAT is NP-complete

$\exists x_1\, \exists x_2\, \exists x_3\, \exists x_4 : (x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$

TQBF is PSPACE-complete

$\exists x_1\, \forall x_2\, \exists x_3\, \forall x_4 : (x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$

configuration c (sequence of variables T_{ia}, H_i, Q_q)

$\Phi_k(c_1, c_2)$ from c_1 to c_2 in (at most) k steps $\quad k \leq 2^{|x|}$

$k = 1$ see SAT construction

$\Phi_{2k}(c_1, c_2) = (\exists c)(\ \Phi_k(c_1, c) \wedge \Phi_k(c, c_2)\ )$ idea OK, but doubles size

$\Phi_{2k}(c_1, c_2) = (\exists c)(\forall c)(\forall c')(\ [\ (c, c') = (c_1, c) \vee (c, c') = (c, c_2)\ ] \rightarrow \Phi_k(c_1, c)\ )$

transparencies made for

Second Course in Formal Languages and Automata Theory

based on the book by Jeffrey Shallit
of the same title

Hendrik Jan Hoogeboom, Leiden
http://www.liacs.nl/~hoogeboo/second/