Theorie-III

Formale Sprachen und Automatentheorie 1+2

examination

Walter Hower

… Recurrence Relation

- illustration: n-ary bit string (vector)
  - incrementing effect
  - adding 1 leading bit ($n{-}1 \xrightarrow{+1} n$) doubles the # input combinations:
    $c_n = 2 \cdot c_{n-1} = 2^1 \cdot 2^{n-1} = 2^{[1+(n-1)]} = 2^n$
    (# rows in a truth table for n bool. var.)
  - example: z_n := # possibilities that ∃! zer0 (*false*)

… Recurrence Relation

$n := 0$ $\boxed{}$ $z_0 = 0$

$n := 1$

$\boxed{0}$ ← (1 x)
$\boxed{1}$ (+ z_0)

$z_1 = 1$

$n := 2$

$\boxed{0}\boxed{0}$
$\boxed{0}\boxed{1}$ ← formerly "1" everywhere (*true*) (1 x)
$\boxed{1}\boxed{0}$ ← formerly ∃! "0" (+ z_1)
$\boxed{1}\boxed{1}$

$z_2 = 2$

… Recurrence Relation

$n := 3$

$\boxed{0}\boxed{0}\boxed{0}$
$\boxed{0}\boxed{0}\boxed{1}$
$\boxed{0}\boxed{1}\boxed{0}$
$\boxed{0}\boxed{1}\boxed{1}$ ← formerly "1" everywhere (1 x)
$\boxed{1}\boxed{0}\boxed{0}$
$\boxed{1}\boxed{0}\boxed{1}$ ← } formerly ∃! "0" (+ z_2)
$\boxed{1}\boxed{1}\boxed{0}$ ←
$\boxed{1}\boxed{1}\boxed{1}$

$z_3 = 3$

⋮

… Recurrence Relation

$z_0 := 0$ ← initial value

$z_{n(>0)} := z_{n-1} + \binom{n-1}{0} = z_{n-1} + 1$

new leading *true* — new leading *false* — recurrence principle

backward substitution: ← development

$= z_{n-2} + 1 + 1 = z_{n-2} + 2$

$= z_{n-3} + 1 + 2 = z_{n-3} + 3$

$\vdots$

$= z_{n-n} + n = z_0 + n = n$

… Recurrence Relation

forward substitution: ← development

$z_1 := z_0 + 1 = 0 + 1 = 1$

$z_2 := z_1 + 1 = 1 + 1 = 2$

$z_3 := z_2 + 1 = 2 + 1 = 3$

$\vdots$

$z_n := n$

… Recurrence Relation

2) d_n := *d*iameter in a grid (q.; n := # *n*odes, … := …)

$d_{\dots}$:= … , $d_{\dots} := d_{\dots -1}$ …

backward substitution: …

forward substitution: …

statement: $d_{\dots}$ = …

proof: induction on …

3) c_h := # *c*onnections in an h-dimensional hyper-cube

… Complexity

- units of cost

– *l*ogarithmic cost: # bits: space complexity

integer ≥ 0: $l(n) := \begin{cases} 1 & ; n \le 1 \\ 1 + \lfloor \mathrm{ld}(n) \rfloor & ; n \ge 2 \end{cases}$

$\mathrm{ld}(n) := \log_2(n)$

– uniform cost: cost(operation) := 1

\# operations (in principle): time complexity

... Complexity

- classes of probl. [formally: languages]
  - $(D)P_{\text{TIME}}$: (deterministically) *p*olynomial ("efficient")

    examples:
    - 2-SAT (max. 2 literals/clause), Horn-SAT (≤ 1 positive lit.) [$\Theta(n)$] ...
    - shortest paths ([Dijkstra: $\Theta(n^2)$], [Floyd-Warshall: $\Theta(n^3)$])
    - Linear Programming ([Khachiyan-1979], [Karmarkar-1984]) (ellipsoid) (interior point)
    - Euler cycle (closed path visiting each *edge* once) ...

... Complexity

- NP_{TIME} : *n*on(-determ.) *p*olyn. ([det.] exponential)

  [*parallel* time compl.: polyn.] $NP_{\text{TIME}} \setminus (D)P_{\text{TIME}} =: AE$ [≠ { } ?]

  [apparently exponential] *

  (inofficial – own definition)

  examples:
  - (job-shop) scheduling [...]
  - Graph Isomorphism [GrIs] [...]
  - Hamilton Cycle [HC] (closed path via each *node* once)

  a given *yes* instance can (determ.) be certified in polyn. time ...

  $(D)P_{\text{TIME}} \subseteq NP_{\text{TIME}}$ $[\Rightarrow NP_{\text{TIME}} \not\subset (D)P_{\text{TIME}}]$!

  $(D)P_{\text{TIME}} \neq NP_{\text{TIME}}$ $[\ (D)P_{\text{TIME}} \subset NP_{\text{TIME}}\ ]$?

* https://www.claymath.org/sites/default/files/***p***vs***np***.pdf

... Complexity

- <u>p</u>olynomial *r*educibility (of *p*robl. by *a*lg., f_r.)
  - $p_1 \le_{\text{p}} p_2$ <u>iff</u> $a_2 \cong f_r^{\text{p}}(a_1)$ similar. via *r*ed. *f*unct. f_r: $w \in L_1 \Leftrightarrow f_r(w) \in L_2$

    (input length) computes difficulty computes

    (Integer LP)
- *NP-complete*ness (TSP, [3-]SAT, ILP, CSP) e.g.
  - (new) problem $p \in NP$ (S. A. Cook's theorem [1971])
  - each probl. in *NP* polynom. reducible to p ... transitive

    (when $p \in P$ could be shown <u>would</u> imply $P = NP$) ...
- *NP-hard*ness (e.g.: k^{th} heaviest subset)
  - $p \in NP$?
  - each problem in *NP* polynom. reducible to p

... Complexity

- *co-NP*$_{\text{TIME}}$
  - introduction
    - HC "complement" [HCc] (<u>no</u> such tour)
    - HCc $\in NP$? (HC $\in NP$)
    - in contrast to P (= $Co\text{-}P \subseteq Co\text{-}NP$)

      A $\in P \to$ Ac $\in P$ $\Rightarrow$

      Ac $\in P \to$ Acc (= A) $\in P$ $\Rightarrow$

      A $\in P \leftrightarrow$ Ac $\in P$ $\Rightarrow$

      (A $\notin P \to$ Ac $\notin P$) $\wedge$ (Ac $\notin P \to$ A $\notin P$)

… Complexity

– illustration

- HCc $\in$ *co-NP* (HC *NP-complete*)
- *NP* $\neq$ *co-NP* ? (HCc $\notin$ *NP* ?)
- complem. of *NP-complete* pr. $\in$ *NP* <u>iff</u> *NP* = *co-NP*
  $\notin$ [?] $\neq$
- [if] $(D)P = NP \xrightarrow[P \text{ closed w.r.t. complement}]{} NP = co\text{-}NP_{\;(=P)}$

  contrapositive: …
- $(D)P\ \ [= Co\text{-}(D)P] \subseteq NP \cap co\text{-}NP$

2 Languages

- Σ (non-empty, finite) alphabet
- Σ^k {strings of length $k_{\,(\geq 0)}$ | symbol $\in \Sigma$}
- $\Sigma^+ := \bigcup_{k:=1}^{\infty} \Sigma^k$
- $\Sigma^* := \bigcup_{k:=0}^{\infty} \Sigma^k = \Sigma^0 \cup \Sigma^+ = \{\varepsilon\} \cup \Sigma^+$ "Kleene closure/star"
- $0 < |\Sigma| < \infty$
- $0 < |\Sigma^k| = |\Sigma|^k < |\Sigma^+| = \omega = 1 + \omega = |\Sigma^*| = \infty$
- type i language: $\exists$ type $i_{\text{max.}}$ grammar $\rightarrow$

…

… Languages

- "type 2" grammar

$G := (S, N, \Sigma, P)$

$S \in N :=$ {non-term.}, $\Sigma :=$ {term.}, $P := \{[N \ni]\ H \longrightarrow z\ [\in (N \cup \Sigma)^*]\}$

variables alphabet := {symbols} *Productions*

$L(G) := \{w \in \Sigma^* \mid S \overset{+}{\underset{G}{\Rightarrow}} w\}$ *context-free*

ex. (G): S, $N := \{S\}$, $\Sigma := \{a, b, +, -, (,)\}$
$P := \{S \rightarrow a, S \rightarrow b, S \rightarrow S+S, S \rightarrow S-S, S \rightarrow (S)\}$
$L(G) = …$ not "regular", but c.-f.

… Languages

- "type 3" (grammars)

$G := (S, N, \Sigma, P)$

$S \in N :=$ {non-t.}, $\Sigma :=$ {t.}, $P := \{[N \ni]\ H \longrightarrow z\ [\in \{\varepsilon\} \cup \Sigma N^{0/1}]\}$

right-linear
…
(no mixture)

$L(G) := \{w \in \Sigma^* \mid S \overset{+}{\underset{G}{\Rightarrow}} w\}$

regular

<u>exercise</u>: L_x = all strings over $\Sigma := \{0,1\}$ with exactly one "0": 1*01*;
…
$L_y = \{a^k b^l \mid k, l \geq 0\}$ $(\Sigma_{\text{ex.}} := \{a, b\})$
…

... Languages

- Chomsky hierarchy

$$|\Sigma^*| =_{\Sigma\neq\emptyset} |\mathcal{N}| \overset{...}{<} 2^{|\mathcal{N}|} = 2^{|\Sigma^*|} \cong |2^{\Sigma^*}|$$

$$\underline{L_{\text{type-3}} \subset L_{\text{type-2}} \subset L_{\text{type-1}}} \subset L_{\text{decidable}} \subset \underline{L_{\text{type-0}}} \subset L_{\text{general}}{}^{:= \{L \mid L \subseteq \Sigma^*\}}$$

$(L\text{-}C)$ ¬ semi-decidable
$\Rightarrow (L, L\text{-}C)$ ¬ decidable

$\boxed{L\text{-}C := \textit{l}\text{anguage-}c\text{omplement}}$

[L "decidable" $\leftrightarrow$ (L "*semi*-decidable" $\wedge$ L-C "*semi*-decidable")]
...
recursively enumerable

$$L_{\substack{\text{prefix-free}\\ \text{regular}}} \subset \left\{ \begin{matrix} L_{\substack{\text{prefix-free}\\ \text{determinist.}\\ \text{context-free}}} \\ L_{\text{type-3}} \end{matrix} \right\} \subset L_{\text{det. c.-f.}} \underset{...}{\subset} L_{\text{type-2}}$$

... Languages

- "word problem" [WP]
  - given
    - G
    - w [$\in \Sigma^*$]
  - decide
    - $w \in L(G)$ [?]

... Languages

– complexity/(un-)computability – of $L(G)$:

- regular ("type 3") : WP $\in$ $(D)\underset{...}{P}$ [$O(|w|)$]
- determ. context-free : WP $\in$ $(D)P$ [$O(|w|)$]
- context-free ("type 2") : WP $\in$ $(D)P$ [$O(|w|^3)$]
- context-sensitive ("type 1") : WP *P-SPACE-complete*
- recursively enumerable ("type 0"): WP not computable [„just" ☺ in general]

3.1 Type 3

- *f*inite *a*utomaton [FA]
  - for regular language[s]/expression[s]
  - *D*FA/*N*FA: *d*eterministic/*n*on-determin. FA
    - DFA (Σ, Q, F, q_0, δ)
      - Σ : [finite] set of [input] symbols
      - Q : [finite] set of states (evtl. incl. *d*ead end)
      - F : [finite] set of *f*inal/accepting states [$\subseteq Q$]
      - q_0 : [initial] start state [$\in Q$]
      - δ : [simple] transition function: $Q \times \Sigma \rightarrow Q$
    - L(DFA) := {$w \in \Sigma^*$ | DFA "accepts" w}

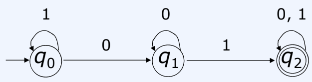
... Type 3

– illustration

- L(DFA) = {p01s | p, s ∈ Σ*}
  - 01, 010, 1001, 00011 ∈ L(DFA)
  - ε, 0, 1, 10, 11, 100 ∉ L(DFA)
  - construction (Σ, q_0, δ, Q, F)
    - » Σ := {0, 1}
    - » q_0 := start state
    - » δ: [exercise ...]
      δ(q_0,) := q_0, δ(q_0,) := q_1,
      δ(q_1,) := q_1, δ(q_1,) := q_2,
      δ(q_2,) := $q_{...}$, ...
    - » Q := {q_0, q_1, q_2}
    - » F := {q_2}

... Type 3

– transition diagram/graph



– transition table

δ	**0**	**1**
→ $\boldsymbol{q_0}$	q_1	q_0
$\boldsymbol{q_1}$	q_1	$q_{2(F)}$
$\boldsymbol{q_{2(F)}}$	$q_{2(F)}$	$q_{2(F)}$

... Type 3

- extended transition function
  $\bar{\delta}(q,w)$ [∈ Q]: Q×Σ* → Q
  - principle of induction [on |w|]
  - recursive construction
    - » $\bar{\delta}(q,\varepsilon)$:= q [|ε| = 0]
    - » w := xa (prefix x ∈ Σ*, 1-char. suffix a ∈ Σ) [∈ Σ⁺]
    - » $\bar{\delta}(q,w)$:= δ($\bar{\delta}(q,x)$, a) [|w| > 0]
    - » L(DFA) := {w ∈ Σ* | $\bar{\delta}(q_0,w)$ ∈ F}

3.2 Type 2

- *p*ush *d*own *a*utomaton (PDA)
  - recognizes context-free language(s)
  - NFA with auxiliary stack
  - *N*PDA: *n*on-deterministic PDA
    - *N*PDA (Σ, Γ, Q, F, q_0, δ) [+ bottom el. ⊥ ...]
      - Σ: [finite] set of input symbols
      - Γ: [finite] set of stack symbols [∋ ⊥, in case]
      - Q: [finite] set of states
      - F: [finite] set of *f*inal/accepting states [⊆ Q]
      - q_0: start state [∈ Q]
      - δ: transition function: Q×Σ*×Γ* → $2^{(Q\times\Gamma^*)}$
    - L(*N*PDA) := {w ∈ Σ* | *N*PDA "accepts" w}

... Type 2

- $L(NPDA)_{\text{empty stack}}$ = $L(NPDA)_{\text{final state(s)}}$
- *D*PDA: *d*eterministic PDA
  - $L(DPDA)_{\text{empty stack}} \subset L(DPDA)_{\text{final state(s)}}$
  - illustration
    - prefix property: $p, s \in \Sigma^+$; $w := ps \in L \Rightarrow p \notin L$ [-free]
    - absence of pref. prop.; ex.: expressions with "(...)"
    - problem: not knowing when p is just a proper prefix

... Type 2

- $L(DPDA) \subset L(NPDA)$
  - ex.: ***r***eversal: $L_r := \{w := w_b \cdot \boldsymbol{r}(w_b) \mid w_b \in \Sigma^*\}$

    [$\Rightarrow |w| = 2k$, $k \in N$ (:= {0, 1, 2, 3, ...}); reversal → palindrome $\not\rightarrow$ r.]
  - corresponding *g*rammar $G_\rho := (S, N, \Sigma, P)$
    $S \in N := \{ \; ... \; \}$, $\Sigma := \{0, 1\}$,
    $P := \{S \rightarrow \varepsilon \mid ... \qquad \}$
    [...]
  - source of *n*on-determinism: no prefix property

... Type 2

- ex.: *m*irror: $\Sigma_m := \Sigma \cup \{m_{\notin\Sigma}\}$

  $L_m := \{w := w_b \cdot m \cdot \boldsymbol{r}(w_b) \mid m \in \Sigma_m \setminus \Sigma, w_b \in \Sigma^*\}$

  [$\Rightarrow |w| = 2k+1$, $k \in N$ (:= {0, 1, 2, 3, ...}); mirror → ¬ rev., r. → ¬ m.]

  prefix-free $L(DPDA)$

  - no guessing concerning the centre
  - the middle ("*m*irror" ☺) is known
  - L_m not regular (¬ type-3)
  - prefix property

... Type 2

- $L(DPDA)_{\text{empty stack}}$ → L prefix-free
- L ¬ prefix-free → $L(DPDA)_{\neg\text{ empty stack}}$
- $L(DPDA)_{\text{empty stack}}$ ↔

  $L(DPDA)_{\text{final state(s)}} \wedge L$ prefix-free

3.4 Type 0

- *T*uring *M*achine (TM)
  - recognizes recursively enumerable lang.
  - *D*TM
    - DTM $(\Sigma, b, M, Q, F, q_0, \delta)$
      - Σ: finite set of input symbols [$\not\ni$ "blank"; $\Sigma \cup \{b\} =: \Sigma_b$]
      - M := {*l*eft, *r*ight, *n*owhere} set of "*move*" actions
      - Q: finite set of states
      - F: finite set of *f*inal/accepting states [$\subseteq Q$]
      - q_0: start state [$\in Q$]
      - δ: transition function: $Q \times \Sigma_b \to Q \times \Sigma_b \times M$
    - $L(\text{DTM}) := \{w \in \Sigma^* \mid \text{DTM "accepts" } w\}$

somewhere else b =: "#" as the separation symbol

4.2 Diagonalization

- general halting problem [Alan M. Turing (1936)]
  - given
    - computer program [algorithm]
    - arbitrary input
  - decide
    - program halts [?]
  - uncomputable
- Georg Ferdinand Ludwig Philipp Cantor (1845 – 1918)

… Diagonalization

"diagonalization" argument, via *halting problem* table:
algorithm A_i halts ("*y*es") or not ("*n*o") on input j

j	0	1	2	3 …
A_0	*y*	*n*	*n*	*y*
A_1	*y*	*n*	*y*	*y*
A_2	*n*	*y*	*n*	*y*
A_3	*y*	*y*	*n*	*y*
⋮				

abbrev.: $\neg n$:= *y*es: termination
$\neg y$:= *n*o termination

$A := (a) :=$

$(\; a_0, \; a_1, \; a_2, \; a_3, \ldots) :=$
$(\neg[A_0,0], \neg[A_1,1], \neg[A_2,2], \neg[A_3,3], \ldots)$
$(\; n, \; y, \; y, \; n, \ldots)$

$\exists i \in \mathcal{N} \; : \; A = A_i$?

no: $\forall i \in \mathcal{N} \; : \; A \neq A_i, \; \exists j_{=i}: a_j \neq [A_i,j]$

$\Rightarrow \forall j \in \mathcal{N} : \exists$ 2 alg. $A, A_k \; : \; a_j \neq [A_{k=j},j]$

$\Rightarrow n \in \{a_j, [A_j,j]\}$ (evtl. *n*o termination)

$\Rightarrow$ $\text{WP}_{\text{type-0}}$ undecidable (uncomputable)

4.3.3 Un-Computability/Un-Decidability

- … Chomsky hierarchy
  - computable ¦ uncomputable !
  - $L_{\text{decidable}}$ ¦ $\subset$ L_0 [$\subset L_{\text{gen.}} := \{L \mid L \subseteq \Sigma^*\}$]
  - L_0 semi-decidable, L_0-C $\neg$ semi-decidable
- $|\Sigma^*| =_{\Sigma \neq \emptyset} |\mathcal{N}| < 2^{|\mathcal{N}|} = 2^{|\Sigma^*|} \cong |2^{\Sigma^*}|$

  $\nexists$ surject.: $\Sigma^* \to 2^{\Sigma^*}$, $\mathcal{N} \to 2^{\mathcal{N}}$ $\exists$ incomparable elements
- $\exists$ only countably infinite many algorithms
- $\exists$ uncountably many languages/probl.
- $\exists$ uncomputable word problems