CSE396 Lecture Thu 4/11 Spring 2019

TopHat #4435

A *K-tape* TM $M = (Q, \Sigma, \Gamma, \delta, \sqcup, s, \{q_{acc}, q_{rej}\})$ has

$\delta \subseteq (Q \setminus \{q_{acc}, q_{rej}\}) \times \Gamma^K \times \Gamma^K \times \{L, R, S\}^K \times Q$.

DTM: δ : " " " $\longrightarrow$ " " "

When $K=2$, a typical instruction is $\left(p, \begin{matrix} c_1 \\ c_2 \end{matrix} / \begin{matrix} d_1 \\ d_2 \end{matrix}, \begin{matrix} D_1 \\ D_2 \end{matrix}, q\right)$ $\quad \textcircled{Q} \begin{matrix} c_1 / d_1, D_1 \\ c_2 / d_2, D_2 \end{matrix}$ $p = q$ allowed

The *input tape* is indexed as "Tape 1".

It is *read-only* if always $d_1 = c_1$ in every instruction.

one-way if D_1 never is L. (S or R are OK).

real time if always $D_1 = R$. (DFAs and NFAs are real-time.

Def^n (replacing notation in text §2.2): A 2-tape TM is a *Pushdown Automaton* (PDA) if

- its input tape is read-only and one-way
- its second tape *behaves as a stack*, meaning: $D_2 = L \Rightarrow d_2 = \sqcup$ in every instruction ("pop")

By the text's normal form, for every state q other than q_{acc}, q_{rej} and sequence $c_1, \dots, c_K$ of chars, δ has *at least one* instruction beginning $\left(q, \begin{matrix} c_1 \\ \vdots \\ c_K \end{matrix} / \dots\right)$.
If there is *exactly* one instruction then M is *deterministic*. (δ is a function).

These defn's carry thru to define *deterministic pushdown automata* (DPDAs) and the default nondeterministic ones (NPDAs). Now we can state:

Theorem: A language $L \subseteq \Sigma^*$ is context-free if and only if there is an NPDA N s.t. $L = L(N)$. (Proof in §2.2, skipped).

Defn: A language L is a *deterministic CFL (DCFL)* if there is a DPDA M such that $L(M) = L$.

Theorem: If L is a DCFL then so is $\tilde{L}$.

$\boxed{\text{But } \{a^i b^j c^k : i \neq j \vee j \neq k\} \text{ is a CFL whose complement is } \underline{\text{not}} \text{ a CFL!}}$

Def^n: (Further than text) A TM M obeys "good housekeeping" if

- it is in the text's normal form with q_{acc}, q_{rej} the only halting states.
- M never writes an actual blank $\sqcup$ between two nonblank chars. Instead it treats '%' as a substitute blank (like %20 for space in URLs). Then its tapes never have internal blanks. (Nor a char after, on right, or before, on left, a blank.)
- whenever M wants to halt, and possibly leave an output $y = f(x)$ on Tape 1 (or some other designated output tape), M erases all other tapes and non-output chars using actual blanks, rewinds to scan the first char of y (or $\sqcup$ if $y = \epsilon$), and finally goes to q_{acc} (or q_{rej}).

If M only wants to define a language $L(M)$, it outputs 1 when $x \in L$, 0 when it rejects an $x \notin L$. (Remember the third possible outcome $M(x)\uparrow$.)

Def^n: A configuration, also called instantaneous description (ID) of a TM M at any point in a computation on an input x, specifies:

- the current state q of M
- the current nonblank tape contents $\vec{w} = (w_1, \ldots, w_k)$ on each tape,

and

- the current head positions $(h_1, \ldots, h_k)$ relative to the start of w_j on each tape j.

We write an ID as $I = \langle q, \vec{w}, \vec{h} \rangle$. When $k=1$ we can code this up as a single string $\langle u q c v \rangle$ where $w = ucv$ over alphabet $Q \cup \Gamma \cup \{\langle, \rangle\}$. (or $c = \sqcup, v = \epsilon, w = u$)

Def^n: Given IDs I and J, write $I \vdash_M J$ ("I can go to J by one step of M") if M has an instruction applicable to I whose execution gives the ID J.

Also write $I \vdash_M^0 I$, $I \vdash_M^k K$ if there is an ID J s.t. $I \vdash_M^{k-1} J$ and $J \vdash_M K$, and $I \vdash_M^* J$ if $I \vdash_M^k J$ for some k.

- 1 tape TM with GH: $I_0(x) = Sx$, $J = q_{acc}1$.

Def^n: $x \in L(M)$ if $I_0(x) \vdash_M^* J$ for some ID J with $q = q_{acc}$, where $I_0(x)$ has $q = S$, x on tape 1, h_1 = first bit of x, all other tapes blank.