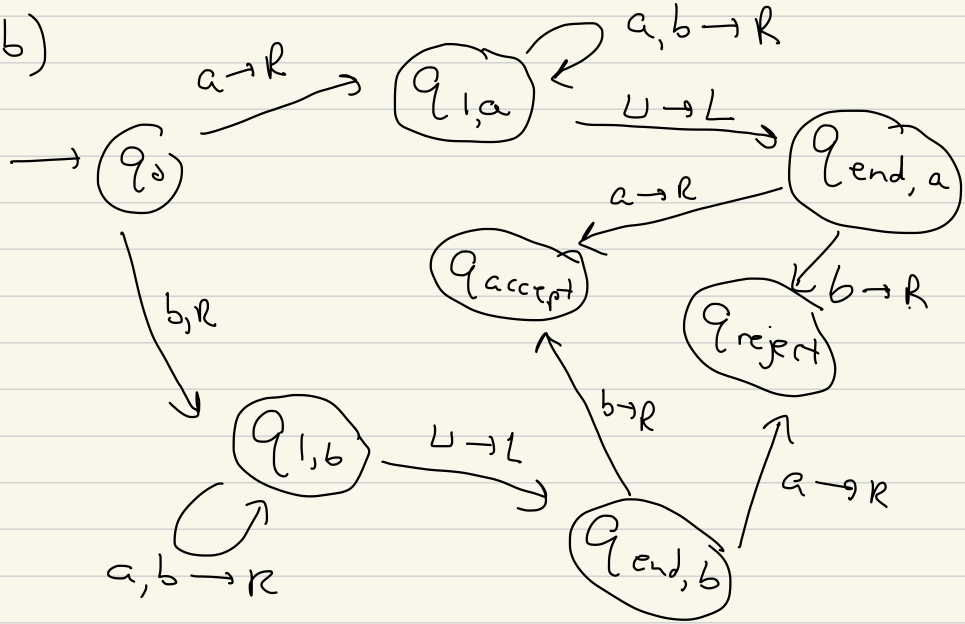
CPSC 421/501 Nov. 6, 2023

Individual Homework 8

(1a) On input s, we remember whether the first symbol is an "a" or a "b" [we do this by having a subset of states to which we transition to upon reading an "a" and a disjoint subset of states upon reading a "b"]. We then accept or reject upon reading the last symbol [on a Turing machine this means that we move right until reaching a blank symbol, ⊔, and then move left one cell].

(1b)



$Q = \{q_0, q_{1a}, q_{end,a}, q_{1b}, q_{end,b}, q_{accept}, q_{reject}\}$

$\Sigma = \{a,b\}$, $\Gamma = \{a, b, \sqcup\}$

δ is described above

$q_0, q_{accept}, q_{reject}$ are $q_0, q_{accept}, q_{reject}$ in Q

(this is a good convention to use)

[There is a lot of flexibility: for example, when transitioning to q_{accept}, the R/L is irrelevant.]

(1c) $q_0 aab$

$a q_{1a} ab$

$a a q_{1a} b$

$a a b q_{1a}$

$a a q_{\text{end }a} b$

$a a b q_{\text{reject}}$

[You can add any finite number of ⊔'s to the right of these configuration descriptions, if you like.]

To see the role of the ⊔ (blank) at the end of the input, you could also write

$q_0 aab\sqcup$

$a q_{1a} ab\sqcup$

$a a q_{1a} b\sqcup$

$a a b q_{1a}\sqcup$

$a a q_{\text{end }a} b\sqcup$

$a a b q_{\text{reject}}\sqcup$

(1d) If $S = \sigma_1 \ldots \sigma_n$ with $\sigma_i \in \{a, b\}$

we begin:

step/config 1: $q_0 \sigma_1 \ldots \sigma_n$

step/config 2: $\sigma_1 q \sigma_2 \ldots \sigma_n$

step/config n: $\sigma_1 \ldots \sigma_{n-1} q \sigma_n$

step/config n+1: $\sigma_1 \ldots \sigma_n q \sqcup$

$q = q_{1a}$ or q_{1b}

step/config n+2: $\sigma_1 \ldots q' \sigma_n \sqcup$ $\quad q' = q_{end,a}$ or $q_{end,b}$

step/config n+3: $\ldots q_{accept}/q_{reject} \ldots$

Total # of steps $= n+3-1 = n+2$

[It is OK to also say n+3 steps, since the last step is step # n+3.]

Group Homework 8

(1) We can recognize L by running a universal Turing machine on $\langle M, \varepsilon \rangle$, accepting $\langle M, \varepsilon \rangle$ if M accepts ε.

To prove that L is undecidable, assume that some Turing machine, P, decides L. Then we claim that we can decide

$$\text{ACCEPTANCE} = \left\{ \langle M, w \rangle \;\middle|\; \begin{matrix} w \text{ accepts} \\ M \end{matrix} \right\}.$$

Indeed, given $\langle M, w \rangle$, we can build a Turing machine P' that constructs M' such that

① M' erases its input

② M' writes w on its input tape, and

③ M' simulates M with these new symbols on its input tape.

④ Having built M', P' now gives P

$\langle M' \rangle$ as input, and therefore checks if $\langle M' \rangle \in L$.

We have
$$M' \in L \iff \langle M, w \rangle \in \text{ACCEPTANCE},$$
and hence P' decides ACCEPTANCE.

This contradicts the fact that ACCEPTANCE is undecidable.

Since L is recognizable but undecidable, the complement of L is unrecognizable.

NOTE: Problem (1) and (2) are essentially the same in the context of Turing machines or in the context of Python programs.

(2) Given $\langle M, q \rangle$ we can tell what is

$\Sigma = \{1, \dots, n_\Sigma\}$, $n_\Sigma \in \mathbb{N}$. Now

write Σ^* in a list:

$w_1, w_2, \dots$

For $k = 1, 2, \dots$, consider an algorithm that

$$\left\{ \begin{array}{l} \text{simulates } \langle M \rangle \text{ for } k \text{ steps on each} \\ \text{of } w_1, \dots, w_k \text{, and we halt and} \\ \text{accept if any configuration enters} \\ \text{state } q \end{array} \right\}$$

If state q of Q is reached by some input, then this algorithm will detect this for some value of k, and therefore accept M.

If not, then this algorithm will never halt. Hence this algorithm recognizes L.

Let us show that L is undecidable:

Say that L is decided by a T.M., P.

Let us show that we can decide ACCEPTANCE: Let P' be the algorithm that builds M' as in Problem 1, and gives $\langle M', q_{accept} \rangle$ as input to L. Then $\langle M', q_{accept} \rangle \in L$ iff M accepts w. Hence this shows that ACCEPTANCE is undecidable, which is impossible.

Hence L is undecidable.

Since L is recognizable but undecidable, the complement of L is unrecognizable.

(3a) $\langle G \rangle$ contains the symbols $\# n_1 \# n_2$ for each edge $\{N_1, N_2\} \in E$, hence at 4 symbols per edge. Hence the length of $\langle G \rangle$, n, is at least $4|E|$. So $n \geq 4|E|$ so $|E| \leq n/4$.

(3b) If $G \in$ HALF-CLIQUE, then for some $|V'| \geq |V|/2 = N/2$, G has at least

$$\binom{|V'|}{2} = \frac{(|V'|)(|V'|-1)}{2} \geq \frac{(N/2)(N/2-1)}{2}$$

$$= \frac{N(N-2)}{8},$$

and hence $\langle G \rangle$ is of length $\geq \frac{N(N-2)}{8}$.

(3c) Consider all subsets $E' \subset E$, of which there are $\leq 2^{n/4}$. For each E', we can form the subset, V', of vertices that are endpoints of the edges in E'.

[For example, if E' is $\{1,2\},\{7,8\},\{1,7\}$, then V' is $\{1,2,7,8\}$]

Then we can check if

1. $|V'| \geq |V|/2$, and
2. if E' consists of all pairs of elements of V'.

For any E', the time needed to

find $\bar{V}'$ and check ①, ② is polynomial in $|E'|$. Since $|E'| \leq |E| \leq n/4$, these operations require time polynomial in $n/4$.

The time it takes to generate all $E' \subset E$ plus this time per each E' is therefore

$$\text{polynomial}(n)\ 2^{n/4}.$$