Quiz #10

Illya Starikov

June 30, 2025

Problem #1

$$
\begin{aligned}
& S \\
\implies & AB && \text{via } S \to AB \\
\implies & aXB && \text{via } A \to aX \\
\implies & aXbYd && \text{via } B \to bYd \\
\implies & aXbYd && \text{via } B \to bYd \\
\implies & abXYd && \text{via } Xb \to bX \\
\implies & abYcd && \text{via } XY \to Yc \\
\implies & abcd && \text{via } Y \to \lambda
\end{aligned}
$$

Problem #2

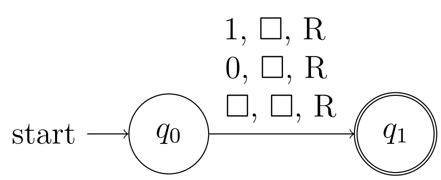
$$
\begin{aligned}
& S \\
\implies & TbC && \text{via } S \to TbC \\
\implies & gC && \text{via } Tb \to g \\
\implies & Sg && \text{via } gC \to Sg \\
\implies & TbCg && \text{via } S \to TbC \\
\implies & gCg && \text{via } Tb \to g \\
\implies & Sgg && \text{via } gC \to Sg \\
\implies & TbCgg && \text{via } S \to TbC \\
\implies & gCgg && \text{via } Tb \to g \\
\implies & egg && \text{via } gC \to e
\end{aligned}
$$

Problem #3

The language is derived by the grammar is $L = \{aaa^n b^k \mid 0 \leq n \leq k, n + k = 2c\}$, for some arbitrary constant c (i.e. $n + k$ is even).

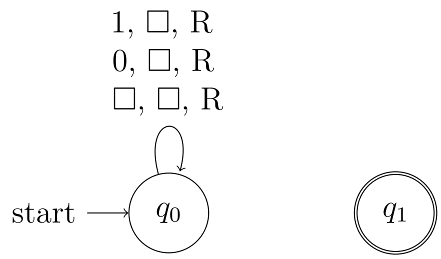
Problem #4

The following always halts, regardless of the input from $\Sigma = \{0, 1\}$.



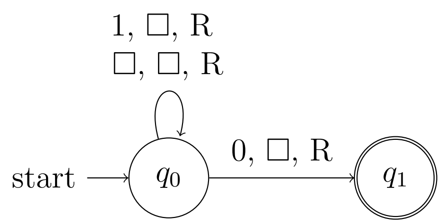
Problem #5

The following never halts, regardless of the input from $\Sigma = \{0, 1\}$. Note it also never halts even if the tape is blank (λ).



Problem #6

The following halts for some, but not all, input from $\Sigma = \{0, 1\}$. In this case, the only input that would cause the machine to halt would be 0.



Problem #7

No, we will prove so as follows.

Theorem 1. *Suppose A to be an arbitrary program — that is, $A \in$ all programs. Then, $\forall A \in$ all program, there* ***does not*** *exists a program P such that P can determine if A will halt* ***regardless of input****.*

Proof. Suppose not. That is, suppose $\exists P \in$ all programs such that P can determine if $\forall A \in$ all programs, A will halt. Suppose the following to be such program P, with an additional source code at the end:

```
bool willHalt(program P) { ··· }

int recursiveNightmare() {
    if (willHalt(P)) {
        return recursiveNightmare();
    }

    return 42;
}
```

If we were to run this program through P, that is $P(P)$, we have the following cases:

Case #1: The program P loops forever. Assuming the program to be correct, this will return false; hence program P halts on input P, which is a contradiction (by our `recursive nightmare`).

Case #2: The program halts on input P. Assuming the program to be correct, this will return true; hence program P runs indefinitely on input P; this is a contradiction.

We see this to be a paradox; P only halts when `willHalt` will not halt, and runs indecently when it does halt. Either way, we see this to be a contradiction. Therefore, there **does not** exists a program P such that P can determine if A will halt **regardless of input**.

$\square$