Non-deterministic Turing machines

Peter Mayr

Computability Theory, January 29, 2021

Generalizations of the DTM

The concept of a Turing machine can be generalized by

1. a tape that is infinite in both directions,
2. a finite number of tapes,
3. **non-determinism**

Notions of configurations, acceptance, language . . . generalize in a straightforward way.

All these generalizations have the same expressive power, i.e., accept the same languages as the DTM defined previously.
(HW)

Definition

- A **non-deterministic Turing machine (NTM)** is a 7-tuple $N = (Q, \Sigma, \Gamma, s, t, r, \Delta)$ like a DTM except for

$$\Delta \colon Q \setminus \{t, r\} \times \Gamma \to P(Q \times \Gamma \times \{-1, +1\}).$$

- (q', α', k') is a **successor configuration** of (q, α, k) in $Q \setminus \{t, r\} \times \Gamma^{\mathbb{N}} \times \mathbb{N}$ if there is $(q', a, d) \in \Delta(q, \alpha(k))$ such that $\alpha' = \alpha[k \to a]$ and $k' = \max(k + d, 0)$.
  Then write $\underline{(q, \alpha, k) \vdash_N (q', \alpha', k')}$.
  Let $\vdash_N^*$ denote the transitive closure of $\vdash_N$.
- N **accepts** $w \in \Sigma^*$ if $(s, w\textvisiblespace\dots, 0) \vdash_N^* (t, ., .)$ for some $(t, ., .)$.
- N **halts** on w if there exists $K \in \mathbb{N}$ such that for each configuration c and $k \in \mathbb{N}$: $(s, w\textvisiblespace\dots, 0) \vdash_N^k c$ implies $k \leq K$.
- $L(N) := \{w \in \Sigma^* \ : \ N \text{ accepts } w\}$ is the **language** of $\cancel{M}$. N

Theorem

For every NTM N there exists a DTM M such that

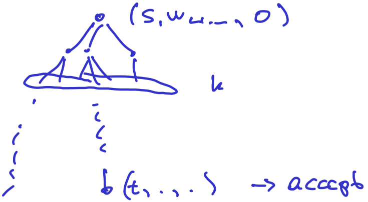
1. $L(N) = L(M)$;
2. if N halts on all inputs, then M halts on all inputs.

Proof.

Let $N = (Q, \Sigma, \Gamma, s, t, r, \Delta)$ be a NTM. The computation of N on input w can be viewed as tree with vertices labelled by configurations where every vertex has at most

$$\max\{|\Delta(q, a)| \ : \ q \in Q \setminus \{t, r\}, a \in \Gamma\}$$

successors.



$w \in L(N)$ iff $\exists$ finite path from root $(s, w\llcorner.., 0)$ to some leaf $(t, ., .)$.

Idea: Construct DTM M that tries all branches of N's computation tree breadth first by determining all configurations C_k that can be reached from the starting configuration in k steps.

High-level description of M:

- Let $C_0 := \{(s, w\_\ldots, 0)\}$ and $k := 0$.
- Iterate the following:
  - If $\exists(t, ., .) \in C_k$, accept.
  - If $C_k = \emptyset$, reject.
  - Let $C_{k+1} := \bigcup_{c \in C_k} \{c' \ : \ c \vdash c'\}$ and $k := k + 1$.

Analysis of M:

- If N accepts w in k steps, then C_k contains an accepting configuration and M will accept (converse is clear).
- If N halts on w in $\leq K$ steps, then either M accepts before reaching C_{K+1} or rejects with $C_{K+1} = \emptyset$. $\square$

Note

If a configuration of the NTM has $\leq r$ successors, then the simulating DTM may consider $|C_k| \leq r^k$ configurations.
It is open whether an exponential increase in the runtime of the DTM can be avoided (see P vs NP).