Lecture #12: Multi-Tape Turing Machines, Nondeterministic Turing Machines, and the Church-Turing Thesis

Key Concepts

Multi-Tape Turing Machines

For any (fixed) positive integer k, a ***k*-tape Turing machine** is a generalization of a Turing machine that has k tapes — all of whose tape heads can move independently. Transitions now allow tape heads to move ***left***, move ***right*** or ***stay*** — so that a k-tape Turing machine can be modelled as a 7-tuple

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}),$$

where Q, Σ, Γ, q_0, q_{accept} and q_{reject} are all as they are for standard Turing machines, and where

$$\delta : Q \times \Gamma^k \to Q \times (\Gamma \times \{\mathtt{L}, \mathtt{R}, \mathtt{S}\})^k$$

is a partial function such that $\delta(q, \sigma_1, \sigma_2, \ldots, \sigma_k)$ is defined whenever $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\sigma_1, \sigma_2, \ldots, \sigma_k \in \Gamma$, and $\delta(q, \sigma_1, \sigma_2, \ldots, \sigma_k)$ is undefined whenever $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\sigma_1, \sigma_2, \ldots, \sigma_k \in \Gamma$.

A ***supplemental document*** (whose first few pages should be examined, as needed) gives more information about how configurations for these Turing machines can be represented, how they process strings, and how the languages of these machines are defined.

The first claim should not be surprising, since any standard Turing machine is a 1-tape Turing machine:

Claim. *Let $\subseteq \Sigma^\star$ (for some alphabet Σ).*

(a) If L is Turing-recognizable then there is a k-tape Turing machine, for some integer $k \geq 1$, that recognizes L.

(b) If L is Turing-decidable then there is a k-tape Turing machine, for some integer $k \geq 1$, that decides L.

A rather complicated ***simulation*** can be used to establish the following too:

Claim. *Let $\subseteq \Sigma^\star$ (for some alphabet Σ). Let k be any integer such that $k \geq 1$.*

(a) If there is a k-tape Turing machine M that recognizes L then there is also a (standard, one-tape) Turing machine $\widehat{M}$ that recognizes L, so that L is Turing-recognizable.

(b) If there is a k-tape Turing machine M that decides L then there is also a (standard, one-tape) Turing machine $\widehat{M}$ that decides L, so that L is Turing-decidable.

A ***multi-tape Turing machine*** is a k-tape Turing machine for some positive integer k.

The above claims imply that the sets of "Turing-recognizable" and "Turing-decidable" languages would not be changed if they were defined using multi-tape Turing machines instead of standard Turing machines.

Nondeterministic Turing Machines

A ***nondeterministic*** Turing machine is the same as a regular Turing machine except that there can be **zero, one, or *many*** moves that might be possible at any time — so that the transition function is now a *total* function

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{\mathtt{L}, \mathtt{R}\}).$$

— such that $\delta(q_{\text{accept}}, \sigma) = \delta(q_{\text{reject}}, \sigma) = \emptyset$ for every symbol $\sigma \in \Gamma$.

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, qR)$ be a nondeterministic Turing machine and let $\omega \in \Sigma^\star$.

- M ***accepts*** ω if *there exists* at least one (finite) sequence of moves, beginning in M's initial configuration for ω, that ends with M in state q_{accept}.
- M ***rejects*** ω if *every* sequence of moves of M, that begins with the initial configuration for ω, is finite — and there are no sequences of moves that end with M in state q_{accept}.
- M ***loops on*** ω otherwise.

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ be a nondeterministic Turing machine and let $L \subseteq \Sigma^\star$.

- M ***recognizes*** L if M accepts every string $\omega \in L$ and M either rejects or loops on every string $\omega \in \Sigma^\star$ such that $\omega \notin L$.
- M ***decides*** L if M accepts every string $\omega \in L$ and M rejects every string $\omega \in \Sigma^\star$ such that $\omega \notin L$ — so that M does not loop on any string in $\Sigma^\star$.[1]

[1] An even *stronger* condition is sometimes required, in order to say that a nondeterministic Turing machine "decides" a language — but adding this does not change the set of languages that are "decidable" by nondeterministic Turing machines.

Claim. *Let $\subseteq \Sigma^\star$ (for some alphabet Σ).*

(a) L is Turing-recognizable if and only if there exists a nondeterministic Turing machine M such that M recognizes L.

(b) L is Turing-decidable if and only if there exists a nondeterministic Turing machine M such that M decides L.

An optional supplemental document includes a sketch of a proof of this claim.

The Church-Turing Thesis

The ***Church-Turing Thesis*** is a widely held belief that Turing machines (and the sets of languages and functions defined using them) really *do* model computability. Additional details about — quite important — thesis are given in a supplemental document about this.