Recall tracing through the configurations of the even machine earlier. The imaginary mechanism consisting of tape, read/write head, and Turing machine program is really just an intuitive way of visualizing what a Turing machine computation is. Formally, we can define the computation of a Turing machine on a given input as a sequence of configurations—and a configuration in turn is a sequence of symbols (corresponding to the contents of the tape at a given point in the computation), a number indicating the position of the read/write head, and a state. Using these, we can define what the Turing machine \( M \) computes on a given input.

**Definition tur.1 (Configuration).** A configuration of Turing machine \( M = \langle Q, \Sigma, q_0, \delta \rangle \) is a triple \( \langle C, m, q \rangle \) where

1. \( C \in \Sigma^* \) is a finite sequence of symbols from \( \Sigma \),
2. \( m \in \mathbb{N} \) is a number < \( \text{len}(C) \), and
3. \( q \in Q \)

Intuitively, the sequence \( C \) is the content of the tape (symbols of all squares from the leftmost square to the last non-blank or previously visited square), \( m \) is the number of the square the read/write head is scanning (beginning with 0 being the number of the leftmost square), and \( q \) is the current state of the machine.

The potential input for a Turing machine is a sequence of symbols, usually a sequence that encodes a number in some form. The initial configuration of the Turing machine is that configuration in which we start the Turing machine to work on that input: the tape contains the tape end marker immediately followed by the input written on the squares to the right, the read/write head is scanning the leftmost square of the input (i.e., the square to the right of the left end marker), and the mechanism is in the designated start state \( q_0 \).

**Definition tur.2 (Initial configuration).** The initial configuration of \( M \) for input \( I \in \Sigma^* \) is

\[
\langle \text{▷} \ I, 1, q_0 \rangle.
\]

The \( \text{▷} \) symbol is for concatenation—the input string begins immediately to the left end marker.

**Definition tur.3.** We say that a configuration \( \langle C, m, q \rangle \) yields the configuration \( \langle C', m', q' \rangle \) in one step (according to \( M \)), iff

1. the \( m \)-th symbol of \( C \) is \( \sigma \),
2. the instruction set of \( M \) specifies \( \delta(q, \sigma) = \langle q', \sigma', D \rangle \),
3. the \( m \)-th symbol of \( C' \) is \( \sigma' \), and
4. a) \( D = L \) and \( m' = m - 1 \) if \( m > 0 \), otherwise \( m' = 0 \), or
   b) \( D = R \) and \( m' = m + 1 \), or
   c) \( D = N \) and \( m' = m \),

5. if \( m' = \text{len}(C) \), then \( \text{len}(C') = \text{len}(C) + 1 \) and the \( m' \)-th symbol of \( C' \) is 0. Otherwise \( \text{len}(C') = \text{len}(C) \).

6. for all \( i \) such that \( i < \text{len}(C) \) and \( i \neq m \), \( C'(i) = C(i) \),

**Definition tur.4.** A *run of \( M \) on input \( I \)* is a sequence \( C_i \) of configurations of \( M \), where \( C_0 \) is the initial configuration of \( M \) for input \( I \), and each \( C_i \) yields \( C_{i+1} \) in one step.

We say that \( M \) *halts on input \( I \) after \( k \) steps* if \( C_k = \langle C, m, q \rangle \), the \( m \)-th symbol of \( C \) is \( \sigma \), and \( \delta(q, \sigma) \) is undefined. In that case, the *output of \( M \) for input \( I \) is \( O \)*, where \( O \) is a string of symbols not ending in 0 such that \( C \overset{\sim}{\rightarrow} O \overset{\sim}{\rightarrow} O' \) for some \( j \in \mathbb{N} \). (\( 0' \) is a sequence of \( j \) 0’s.)

According to this definition, the output \( O \) of \( M \) always ends in a symbol other than 0, or, if at time \( k \) the entire tape is filled with 0 (except for the leftmost \( \triangleright \)), \( O \) is the empty string.

**Photo Credits**

**Bibliography**