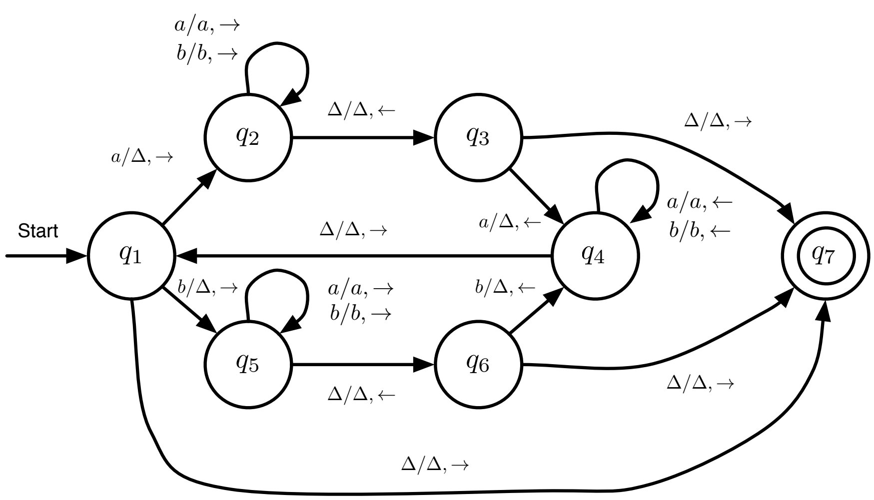
CS 360
Naomi Nishimura

Turing machines

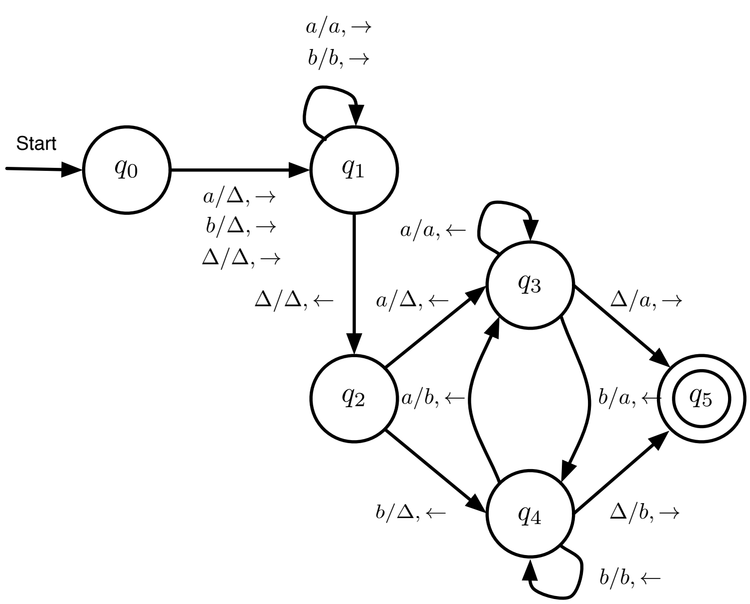
1 Turing machine example

The following transition diagrams are of Turing machines discussed in class.

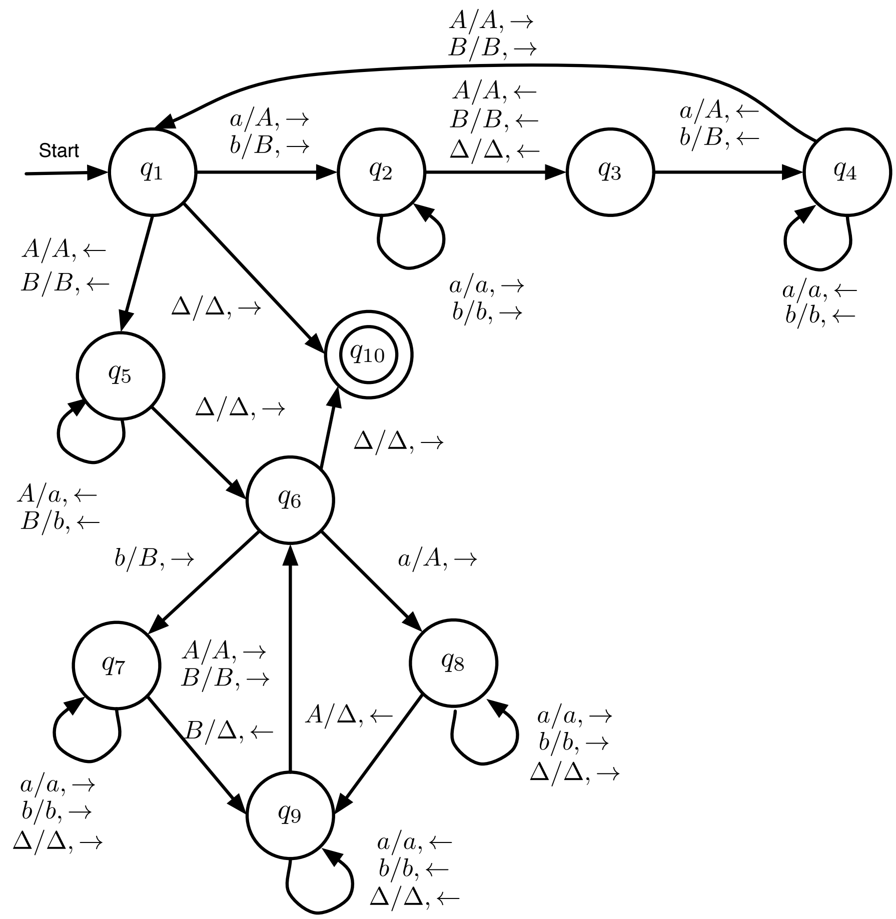
A Turing machine for L_{PAL}:



A Turing machine that deletes the current symbol:



A Turing machine for $\{ss \mid s \in \{a,b\}^*\}$:



2 Turing-computability and decidability

A partial function $f : \Sigma^* \rightarrow \Gamma^*$ is *Turing-computable* if it is computed by a one-tape deterministic Turing machine, that is, if for every x on which f is defined, $q_0 x \vdash^* h_a f(x)$ and on any other input, T fails to halt. This can be generalized to a partial function $f : (\Sigma^*)^k \rightarrow \Gamma^*$, where f is Turing-computable if on every $(x_1, \ldots, x_k)$ on which f is defined, $q_0 x_1 \Delta x_2 \Delta x_3 \ldots \Delta x_k \vdash^* h_a f(x_1, \ldots, x_k)$.

A language is *Turing-decidable* if its characteristic function is Turing-computable, where the characteristic function for L, χ_L, is defined as $\chi_L(x) = 1$ if $x \in L$ and $\chi_L(x) = 0$ otherwise.

3 Nondeterministic multitape machine for composites

The language is the set $\{1^n \mid n \text{ is composite}\}$. We consider two methods, detailed below; the first uses the fact that a composite number n is divisible by some integer in the range from 2 to $n-1$ and the second uses the fact that a composite number is the product of two integers in the range from 2 to $n-1$.

3.1 Division approach

Here we use Tape 1 to store the input and Tape 2 to store our "guess" 1^p, where we wish to show that p divides n.

We can divide the machine into subtasks as follows (many details omitted):

- Guess p: Write two 1's, then nondeterministically choose between adding a 1 and going to the next step.
- Divide: Repeatedly match Tapes 1 and 2, erasing 1's from Tape 1 (reject if $p > n$).
- Check: Accept if after a complete iteration of Divide only blanks remain on Tape 1.

For a deterministic machine, instead of guessing p, each possible value of p can be tried in turn. An additional subtask would be used to increment the value.

3.2 Multiplication approach

For this approach, again Tape 1 is used to store the input. Tapes 2 and 3 store our "guesses" of 1^p and 1^q, where we wish to show that $pq = n$. Tape 4 is used to store pq.

We use the following subtasks (details omitted):

- Guess p
- Guess q
- Multiply p and q
- Compare Tapes 1 and 4