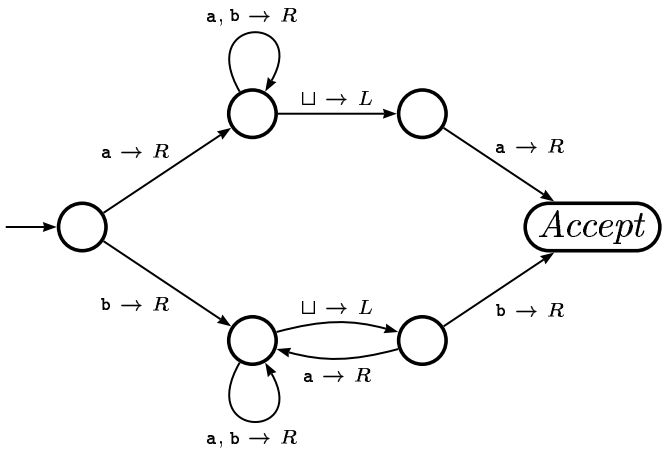
Language Deciders vs. Language Recognizers

Every TM T partitions the set of possible input strings over its input alphabet into three sets:

- $\textsc{Accept}(T)$ = the set of input strings for which T halts by reaching its accept state;
- $\textsc{Reject}(T)$ = the set of input strings for which T halts by reaching its reject state; and
- $\textsc{Loop}(T)$ = the set of input strings for which T never halts.

For example, consider the following TM M for strings over the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$:



It is easily checked that

- $\textsc{Accept}(M)$ = all strings of the form $\mathtt{a}\Sigma^*\mathtt{a} \cup \mathtt{b}\Sigma^*\mathtt{b} \cup \Sigma$;
- $\textsc{Reject}(M)$ = all strings of the form $\mathtt{a}\Sigma^*\mathtt{b} \cup \varepsilon$; and
- $\textsc{Loop}(M)$ = all strings of the form $\mathtt{b}\Sigma^*\mathtt{a}$.

Using this terminology, we can redefine recognizers and deciders as follows:

- A TM T is a *recognizer* for a language L if $\textsc{Accept}(T) = L$.
- A TM T is a *decider* for a language L if $\textsc{Accept}(T) = L$ and $\textsc{Loop}(T) = \Phi$. Equivalently, a TM T is a decider for L if $\textsc{Accept}(T) = L$ and $\textsc{Reject}(T) = \overline{L}$.

We see that the above example M is a recognizer for the language $\mathtt{a}\Sigma^*\mathtt{a} \cup \mathtt{b}\Sigma^*\mathtt{b} \cup \Sigma$ but it is not a decider for this language.

Recall these definitions:

- A language L is *Turing-recognizable* if there is some TM that recognizes it.
- A language L is *decidable* if there is some TM that decides it.

We will show that there are languages that are: (1) Turing-recognizable but not decidable, and (2) languages that are not even Turing-recognizable. A language in the first category has the peculiar property that any TM that recognizes it must fail to terminate for some input strings. A language in the second category has the even more peculiar property that there is *no* TM that accepts exactly those strings belonging to that language. (By the Church-Turing thesis, this means that there is no *algorithm* that is able to return with an *accept* result for exactly those strings belonging to such a language.)