4. **Problem**: Extend the notion of a Turing machine by providing the possibility of a two-way infinite tape. Show that nevertheless such machines recognize exactly the same languages as the standard machines whose tape is only one-way infinite.

**Solution**: A Turing machine with a two-way infinite tape works otherwise in a same way than a standard machine except that the position of the tape start symbol (>\textsuperscript{0}) is not fixed and it can move in a same way than the end symbol (<\textsuperscript{0}). The tape positions are indexed by the set $\mathbb{Z}$ of integers where 0 denotes the initial position of >\textsuperscript{0}.

We can simulate such a Turing machine by a two-track one-way Turing machine. Conceptually, we divide the tape into two parts: upper and lower. The upper part holds the two-way tape cells $i \geq 0$ and the lower part cells $i < 0$. For example, a two-way tape:

\[
\varepsilon > a \ v \ b \ a \ b \ a \ < \ v \ \varepsilon
\]

is expressed as a one-way tape:

\[
\begin{array}{cccccc}
\varepsilon & > & a & b & a & < \textsuperscript{'} \ \\
\varepsilon & b & > & \varepsilon & \varepsilon & < \textsuperscript{'}
\end{array}
\]

In practice the construction of two tracks is done by replacing the alphabet $\Sigma$ by a new alphabet $\Sigma' = (\Sigma \cup \{<\textsuperscript{',}>,\textsuperscript{'}\}) \times (\Sigma \cup \{<\textsuperscript{',}>,\textsuperscript{'}\})$. Each symbol of $\Sigma'$ thus denotes two symbols of $\Sigma$. The symbols $\{<\textsuperscript{',}>,\textsuperscript{'}\}$ are new symbols that denote the start and end symbols of the original tape. So, the above example is expressed as:

\[
\begin{array}{cccccc}
> & (a, b) & (b, >) & (a, \varepsilon) & (\varepsilon, \varepsilon) & <
\end{array}
\]

We still need a way to decide which tape-half is used. The easiest way to do this is to define a mirror image state $q'$ for each state $q$. When the machine is in state $q$, it examines only the upper track when it decides what move to take next (tape head is on right side of the tape). Similarly, in state $q'$ it examines only the lower symbol (tape head is on the left side). Since the lower tape is in a reversed order, all tape head moving instructions have to be also reversed.

The formal definition of this construction is presented in an appendix.

5. **Problem**: Show that Turing machines whose tape alphabet contains at most two symbols in addition to the input symbols are capable of recognising exactly the same languages as the standard machines.

**Solution**: Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$ be a Turing machine such that $|\Gamma - \Sigma| > 2$. We want to construct a machine $M'$ such that $|\Gamma' - \Sigma| = 2$. Let $\Gamma = \{a_1, \ldots, a_n\}$. The basic idea of the construction is to identify the elements of $\Gamma$ with the integers $\{1, \ldots, n\}$ and represent them as $k$-bit integers, where $k = \lceil \log_2(|\Gamma|) \rceil$. In other words, each element of $M$'s tape
The transition function \( M \) of a Turing machine \( M \) is defined so that for each step of \( M \), \( M \) does first \( k \) steps where it first decides what symbol of \( \Gamma \) is encoded in the tape cells to the right of the read/write head. This can be done using a Turing machine that reads \( k \) symbols from the tape while moving its head to right at each step and that remembers the input in its states. For example, if \( k = 3 \), then the following Turing machine may be used:

If the machine ends in the state 011, then the input symbol is \( a_3 \) since \( 011_2 = 3_{10} \). The symbol that is written to the tape is similarly done using \( k \) different transitions. Finally, the tape head is moved \( k \) steps to the correct direction.

**Appendix: the formalisation of solution 4**

Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}}) \) be a two-way tape Turing machine. Define a standard Turing machine \( M' \) as follows:

\[
M' = (Q', \Sigma', \Gamma', \delta', q_0, q_{\text{acc}}, q_{\text{rej}})
\]

where \( Q' = Q \cup \{ q' \mid q \in Q \} \)

\( \Sigma' = (\Sigma \cup \{ <', '>' \}) \times (\Sigma \cup \{ <', '>' \}) \)

\( \Gamma' = (\Gamma \cup \{ <', '+' \}) \times (\Gamma \cup \{ <', '+' \}) \)

The transition function \( \delta' \) is defined as follows:

\[
\delta' = \{ (q_1, (a, \gamma), q_2, (b, \gamma), \Delta) \mid (q_1, a, q_2, b, \Delta) \in \delta, \gamma \in \Gamma' \} \\
\cup \{ (q_1, (\sigma', \gamma), q_2, (b, \gamma), \Delta) \mid (q_1, \sigma, q_2, b, \Delta) \in \delta, \gamma \in \Gamma', \sigma \in \{ <, > \} \} \\
\cup \{ (q_1', (\gamma, a), q_2', (\gamma, b), \overline{\Delta}) \mid (q_1, a, q_2, b, \Delta) \in \delta, \gamma \in \Gamma' \} \\
\cup \{ (q_1, (\gamma, \gamma), q_2', (\gamma, b), \overline{\Delta}) \mid (q_1, a, q_2, b, \Delta) \in \delta, q_{\text{end}} \in \{ q_{\text{acc}}, q_{\text{rej}} \}, \gamma \in \Gamma' \} \\
\cup \{ (q_1, (\gamma, \gamma'), q_2', (\gamma, b), \overline{\Delta}) \mid (q_1, \sigma, q_2, b, \Delta) \in \delta, \gamma \in \Gamma', \sigma \in \{ <, > \} \} \\
\cup \{ (q_1, (\gamma, \gamma'), q_2, (b, \gamma), \overline{\Delta}) \mid (q_1, \sigma, q_2, b, \Delta) \in \delta, \gamma \in \Gamma' \} \\
\cup \{ (q_1, (\gamma, \gamma'), q_2, (\gamma, b), \overline{\Delta}) \mid (q_1, \sigma, q_2, b, \Delta) \in \delta, \gamma \in \Gamma', \sigma \in \{ <, > \} \} \\
\cup \{ (q_1, (\gamma, \gamma'), q_2, (\gamma, b), \overline{\Delta}) \mid (q_1, \sigma, q_2, b, \Delta) \in \delta, \gamma \in \Gamma' \}
\]

where \( \overline{\Delta} = R, \overline{L} = L, \overline{\gamma} = > \) and \( \overline{\gamma} = < \).