Lecture 14. Turing Machines

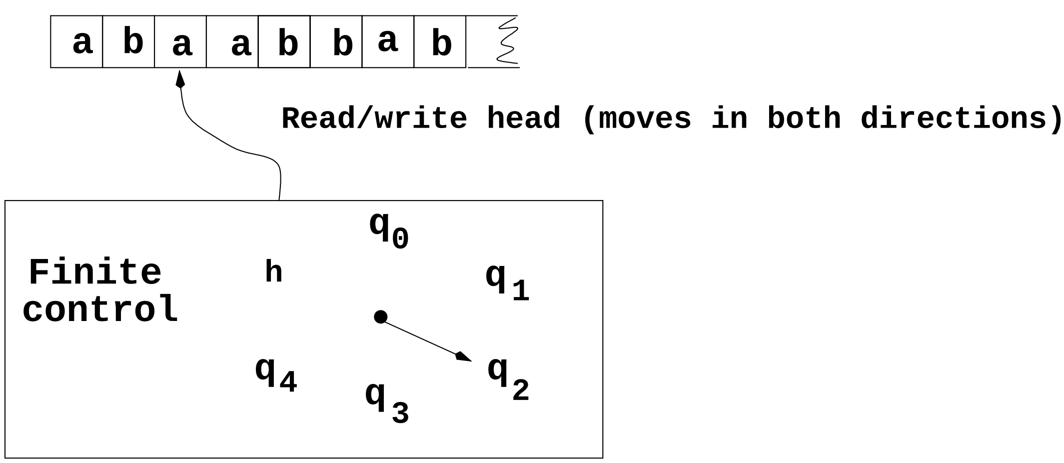
Outline

- (Deterministic) Turing machines
- How TMs work
- How TMs are specified
- Computations for TMs

- Pushdown automata are too restrictive (uses stack) to serve as models of general purpose computers.
- TMs have unlimited and unrestricted memory (allowing writing to tape and reading back the stored information).
- TM can do everything that a real computer can do.
- We shall see that even TM cannot solve certain problems; these problems are beyond the theoretical limits of computation.

TM is a formal model of computers, consisting of

- A tape with a left end, but extends indefinitely to the right,
- A finite control with a finite number of states, and
- A read/*write* head.



First example of Turing Machines

How does a TM recognize the following non-context-free language

$$\{w\#w | w \in \{0,1\}^*\}?$$

M: On input string w:

1. zig-zag across the tape to corresponding positions on both sides of the # symbol to check whether these positions contain the same symbol. If they do not, or if no # is found, reject w. Else cross off matched symbols.

2. When all symbols to the left of # have been crossed off, check for any remaining symbols to the right of #. If any symbols remain, reject w, otherwise accept w.

Basic Operations of Turing Machines

- At the beginning of computation, assume that the left end of tape has a special symbol $\triangleright$; the input string is placed just to the right of $\triangleright$, the rest of the tape is blank ($\sqcup$).
- At each step, the machine reads the symbol that the R/W head is currently pointing to, and depending on its current state and the symbol read.
  1. Enter the next state,
  2. Do one of the following (but not both):
     - Write a symbol in the current tape square, or
     - Move R/W head one tape square to the left or right.
- Computation stops when it enters a *halting state*. The machine might have read only part of the string.
- The symbols left on the tape when it halts is the *output*.
- TMs are deterministic.

Note: The initial position of the head is often assumed to be immediately to the right of $\triangleright$, but we can specify it to be anywhere.

Formal definition

A Turing machine is a quintuple $(K, \Sigma, \delta, s, H)$ where

- K — a finite set of **states**,
- Σ — **alphabet**, ($\sqcup, \triangleright \in \Sigma$, but $\rightarrow$, $\leftarrow \notin \Sigma$),
- $s \in K$ — the **initial state**,
- $H \subseteq K$ — a set of **halting states**,
- δ — **transition function**

$$(K - H) \times \Sigma \quad \rightarrow \quad K \times (\Sigma \cup \{\leftarrow, \rightarrow\})$$

  $\delta(q, a) = (p, b)$ means if M is in state q and read a, then M will enter state p and

  - writes b (overwrite a), if $b \in \Sigma$,
  - moves the tape head if b is $\leftarrow$ or $\rightarrow$

Note:

- M is deterministic since δ is a function (not a relation).
- δ is not defined for the states in H, i.e. the computation does not continue when the machine enters a halting state.

Assumptions

- Whenever the tape head reads $\triangleright$, it immediately moves to the right, i.e. the head does not fall off the left end, and $\triangleright$ is never erased.

  For all $q \in K - H$,
  if $\delta(q, \triangleright) = (p, b)$, then $b = \rightarrow$.

- Never write $\triangleright$ on the tape, i.e. $\triangleright$ always indicate left end of tape.

  For all $q \in K - H$ and $a \in \Sigma$,
  if $\delta(q, a) = (p, b)$, then $b \neq \triangleright$.

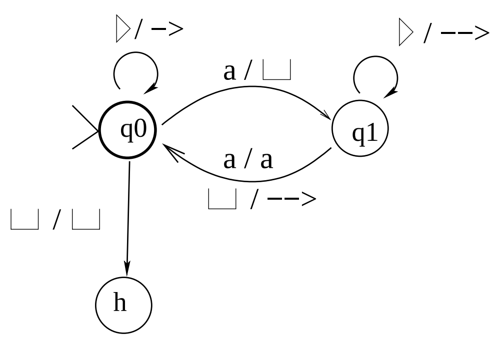
Examples

Example 1: Change all a's to $\sqcup$'s until it encounters $\sqcup$.

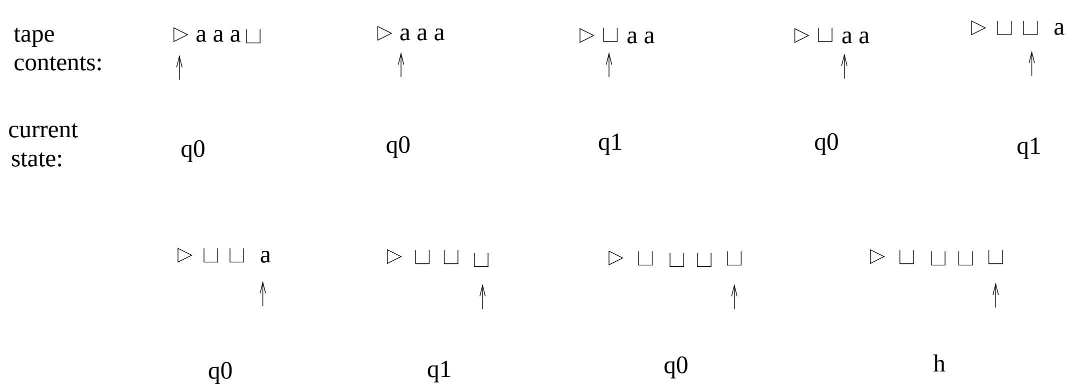
$$M = (K, \Sigma, \delta, s, \{h\})$$

- $K = \{q_0, q_1, h\}$, $\Sigma = \{a, \sqcup, \triangleright\}$, $s = q_0$.
-

δ	a	$\sqcup$	$\triangleright$
q_0	$(q_1, \sqcup)$	$(h, \sqcup)$	$(q_0, \rightarrow)$
q_1	(q_0, a)	$(q_0, \rightarrow)$	$(q_1, \rightarrow)$



M alternate between states q_0 and q_1, replacing an a with a blank $\sqcup$. Note that $\delta(q_1, a)$ is useless since M can never be in state q_1 while scanning an a if it is started in state q_0. Nevertheless, since TMs are deterministic, we need to define $\delta(q, \sigma)$ for all values of $K - H \times \Sigma$.



Examples

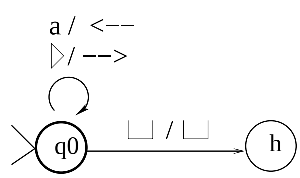
Example 2: Here is a trivial TM to demonstrate that some TMs may loop forever.

$$M = (K, \Sigma, \delta, s, \{h\})$$

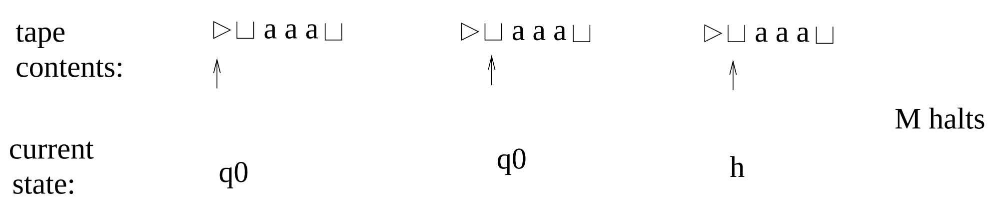
where

$$K = \{q_0, h\},\ \Sigma = \{a, \sqcup, \triangleright\},\ s = q_0,\ H = \{h\},$$

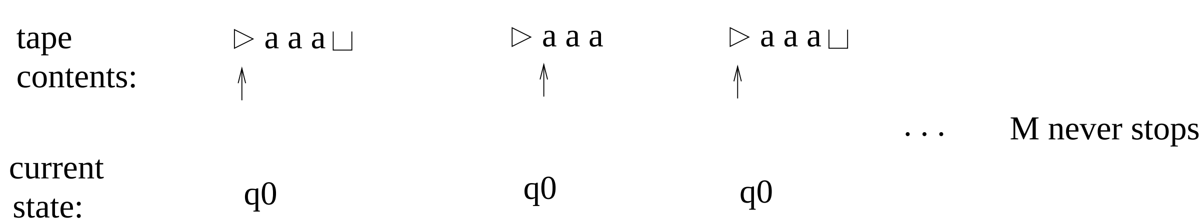
δ	a	$\sqcup$	$\triangleright$
q_0	$(q_0, \leftarrow)$	$(h, \sqcup)$	$(q_0, \rightarrow)$



Starting from the left end marker, the machine scans to the right, if it sees a blank, it halts.

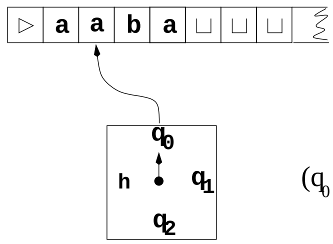


If it sees an a, then M will move left, and from then on it will indefinitely go back and forth between the two squares.



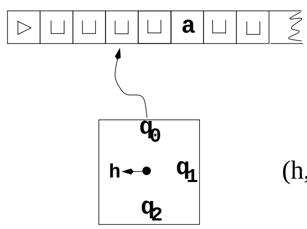
Configuration

Configuration: for specifying the status of a TM computation. (current state, current tape content with the tape head position indicated), e.g. $(q_0, \triangleright \sqcup a\underline{b}a)$.



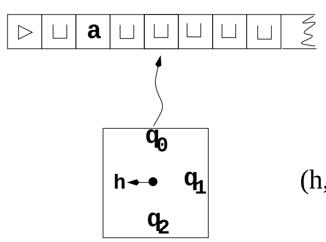
$(q_0, \triangleright\ a\,\underline{a}\,b\,a)$

* blanks to the right of string are not represented.



$(h, \triangleright \sqcup\sqcup\underline{\sqcup}\sqcup\ a)$

* blanks to the left of string are represented.



$(h, \triangleright \sqcup a \sqcup \underline{\sqcup})$

* blanks to the right of string are represented if the tape head move beyond the string.

Configuration

Let C_i be configurations.

- if M can go from C_1 to C_2 in a single step, we write

$$C_1 \vdash_M C_2$$

- Computation by M

$$C_0 \vdash_M C_1 \vdash_M C_2 \vdash_M \cdots \vdash_M C_n$$

- $\vdash_M^*$: yields in 0, 1 or more steps.
- $\vdash_M^n$: yields in n steps.

Configuration

Example: refer to the TM in Example 1.

$$M = (K, \Sigma, \delta, s, \{h\})$$

where $K = \{q_0, q_1, h\}$, $\Sigma = \{a, \sqcup, \rhd\}$, $s = q_0$.

δ	a	$\sqcup$	$\rhd$
q_0	$(q_1, \sqcup)$	$(h, \sqcup)$	$(q_0, \rightarrow)$
q_1	(q_0, a)	$(q_0, \rightarrow)$	$(q_1, \rightarrow)$

Computation:
$(q_0, \underline{\rhd}aaa) \vdash (q_0, \rhd\underline{a}aa) \vdash (q_1, \rhd\underline{\sqcup}aa) \vdash (q_0, \rhd \sqcup \underline{a}a) \vdash$

$(q_1, \rhd \sqcup \underline{\sqcup}a) \vdash (q_0, \rhd \sqcup \sqcup\underline{a}) \vdash (q_1, \rhd \sqcup \sqcup\underline{\sqcup}) \vdash$

$(q_0, \rhd \sqcup \sqcup \sqcup \underline{\sqcup}) \vdash (h, \rhd \sqcup \sqcup \sqcup \underline{\sqcup})$

- TMs in the tabular form is complex and hard to interpret.
- Need a hierarchical graphical notation
  - Build complex machines from simpler ones
  - Arrows denote transitions, but connecting TMs, rather than states.

Basic machines

1. Symbol-writing machines:
   M_a: writes a in the current square (no matter what is read from there) and halts.

   $M_a = (\{s,h\}, \Sigma, \delta, s, \{h\})$.
   $\delta(s,\sigma) = (h,a)$, for any $\sigma \in \Sigma - \{\triangleright\}$,
   $\delta(s,\triangleright) = (s,\rightarrow)$.

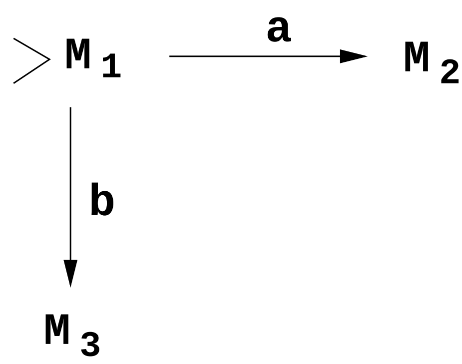
   For each $\sigma \in \Sigma - \{\triangleright\}$, define M_σ.

2. Head-moving machines:
   $M_\rightarrow$: moves head one square to the right and halts.

   $M_\rightarrow = (\{s,h\}, \Sigma, \delta, s, \{h\})$.
   $\delta(s,\sigma) = (h,\rightarrow)$, for any $\sigma \in \Sigma - \{\triangleright\}$,
   $\delta(s,\triangleright) = (s,\rightarrow)$.
   $M_\leftarrow$ is defined similarly.

M_a is abbreviated as a.
$M_\rightarrow$ and $M_\leftarrow$ are abbreviated as R and L, respectively.

Combining TMs



If M_1, M_2, and M_3 are TMs, then the above machine operates as follows:

- Start at the initial state of M_1, simulate the operation of M_1 till M_1 halts.
- If the currently scanned symbol is a, run M_2 with the output of M_1 as the input to M_2 and the final head position of M_1 as the initial head position of M_2. If M_2 halts, then the composite machine halts. The output of the composite machine is that of M_2.
- Otherwise if the currently scanned symbol is b, run M_3 with the output of M_1 as the input to M_3 and the final head position of M_1 as the initial head position of M_3. If M_3 halts, then the composite machine halts. The output of the composite machine is that of M_3.

Combining TMs

It is clear that a composite TM can be explicitly defined using the definitions of its constituents:

$M_1 = (K_1, \Sigma, \delta_1, s_1, H_1)$
$M_2 = (K_2, \Sigma, \delta_2, s_2, H_2)$
$M_3 = (K_3, \Sigma, \delta_3, s_3, H_3)$
Assume K_1, K_2, K_3 are disjoint.

Composite machine: $M = (K, \Sigma, \delta, s, H)$,
where $K = K_1 \cup K_2 \cup K_3$, $s = s_1$, $H = H_2 \cup H_3$.

For each $\sigma \in \Sigma$, $q \in K - H$, need to define $\delta(q, \sigma)$:

- If $q \in K_1 - H_1$, $\delta(q, \sigma) = \delta_1(q, \sigma)$
- If $q \in K_2 - H_2$, $\delta(q, \sigma) = \delta_2(q, \sigma)$
- If $q \in K_3 - H_3$, $\delta(q, \sigma) = \delta_3(q, \sigma)$
- If $q \in H_1$,

$$\begin{aligned}
\delta(q, a) &= (s_2, a) \\
\delta(q, b) &= (s_3, b) \\
\delta(q, \sigma) &= (h, \sigma) \text{ where } h \in H_2 \cup H_3 \qquad \text{if } \sigma \neq a, b
\end{aligned}$$

Graphical notation

1) $> R$ Move head right one square

2) $> R \longrightarrow a \equiv Ra$

Move head right one square, then write a

3) $> R \xrightarrow{a,\, b,\, \sqcup,\, \triangleright} R \equiv \; > R \longrightarrow R \equiv RR \equiv R^2$

if $\Sigma = \{ a, b, \sqcup, \triangleright \}$

Move head right one square; then, if that square contains an a, b, $\sqcup$, or $\triangleright$ moves head one square further to the right.

4)

$\Sigma = \{ a, b, c, \sqcup, \triangleright \}$

$$
\begin{tikzcd}
> R \arrow[loop above, "\sqcup"] \arrow[rr, "\sigma \neq \sqcup"] & & L\,\sigma
\end{tikzcd}
\equiv
\begin{tikzcd}
> R \arrow[loop above, "\sqcup"] \arrow[rr, "a"] \arrow[drr, "b"] \arrow[dr, "c"'] & & L\,a \\
 & L\,c & L\,b
\end{tikzcd}
$$

Scans to the right until it finds a nonblank square, then copies the symbol on to the square immediately to the left of where it was found.

Graphical notation

4) $$> R \circlearrowleft^{\overline{\sqcup}} \quad \equiv \quad R_{\sqcup}$$

Finds the first blank square to the right of currently scanned square.

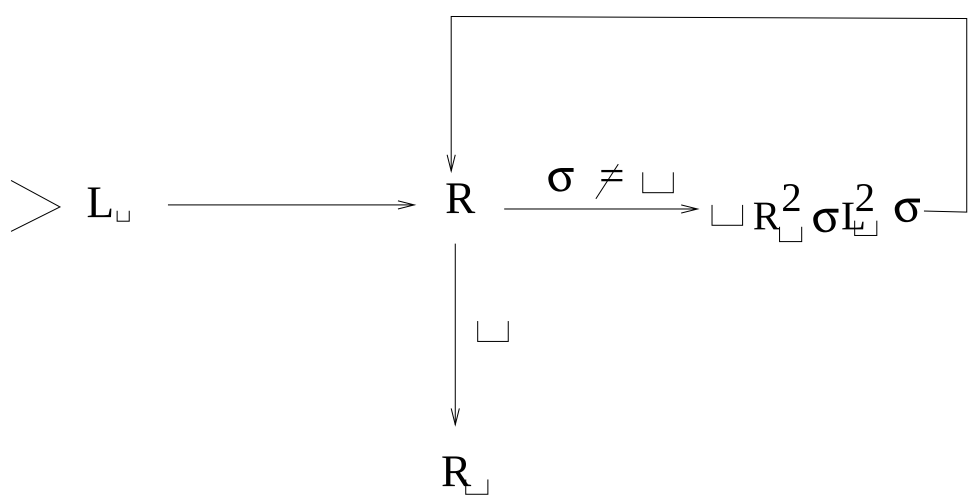
$R_{\overline{\sqcup}}$

$L_{\sqcup}$

$L_{\overline{\sqcup}}$

Copying machine

Assume the input string w contains only nonblank symbols. Transform $\triangleright \sqcup w \underline{\sqcup}$ into $\triangleright \sqcup w \sqcup w \underline{\sqcup}$. (note here we specify the tape head as starting at a position other than immediately to the right of $\triangleright$).



1. Scan left until the first blank symbol is encountered, then move right. ($L_\sqcup \to R$)

2. Remember the current symbol σ (using the finite control), and replace it by a blank symbol.

3. Move right to the very end of the tape content and pass the first blank ($R^2_\sqcup$) and write the remembered symbol σ there.

4. Scan left until the second blank is encountered (i.e. where the last copied symbol was blanked out) and write back that symbol ($L^2_\sqcup\sigma$). Then move right and repeat the process.

$\triangleright \sqcup abb\sqcup$

$\triangleright \sqcup \sqcup bb \sqcup a$

$\triangleright \sqcup abb \sqcup a$

$\triangleright \sqcup a \sqcup b \sqcup a$

$\triangleright \sqcup a \sqcup b \sqcup ab$

$\triangleright \sqcup abb \sqcup ab$

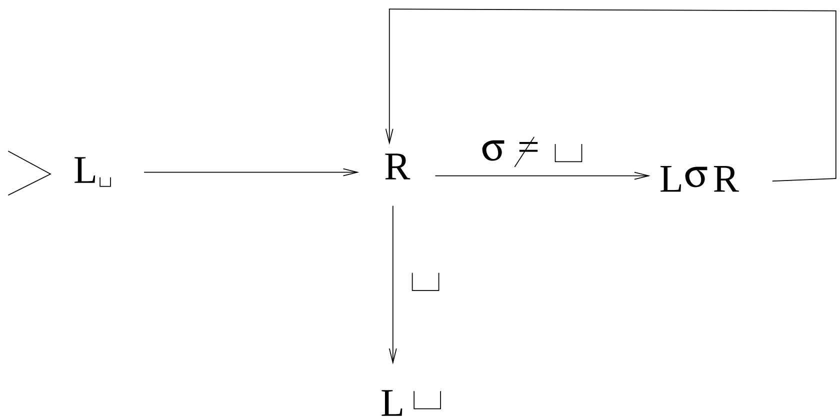
$\triangleright \sqcup ab \sqcup \sqcup ab$

$\triangleright \sqcup ab \sqcup \sqcup abb$

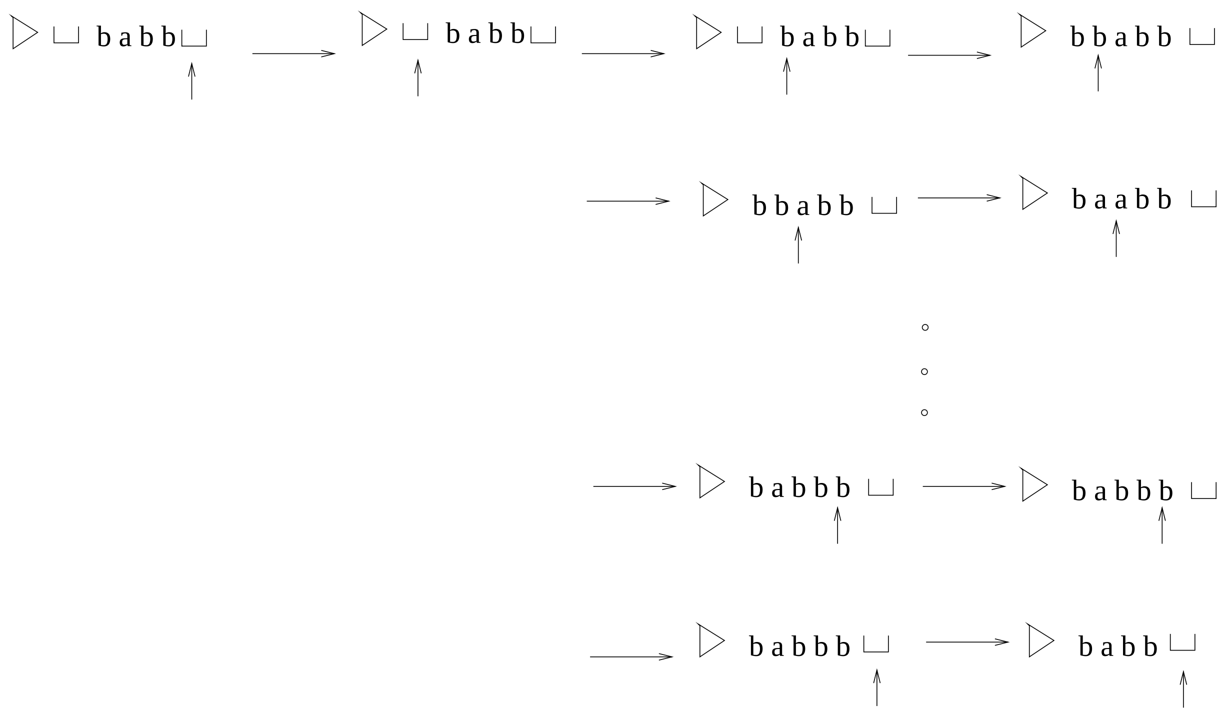
$\triangleright \sqcup abb \sqcup abb$

Left-shifting machine $S_{\leftarrow}$

Transform $\rhd \sqcup w \underline{\sqcup}$ into $\rhd w \underline{\sqcup}$ (error in text book).



1. Scan left until the first blank symbol is encountered, then move right. ($L_{\sqcup} \rightarrow R$)
2. If the current symbol is not a blank, write that symbol to the left of current position ($L\sigma$), go to 4.
3. Else if the current symbol is a blank, write a blank to the left of current position and stop ($L\sqcup$).
4. Move right to the next symbol and repeat the process.


b a b b
b a b b
b a b b
b b a b b
b b a b b
b a a b b
b a b b b
b a b b b
b a b b b
b a b b