

Week 8 Tutorial  
Lambda Calculus and Turing Machines  
Solution

---

## 1 Warm Up

### 1.1

$$\begin{aligned} & \underline{(\lambda f. f (f 5)) (\lambda m. m + 1)} \\ & \longrightarrow (\lambda m. m + 1) \underline{((\lambda m. m + 1) 5)} \\ & \longrightarrow (\lambda m. m + 1) \underline{(5 + 1)} \\ & \longrightarrow \underline{(\lambda m. m + 1) 6} \\ & \longrightarrow 6 + 1 \\ & \longrightarrow 7 \end{aligned}$$

### 1.2

$$\begin{aligned} & \underline{(\lambda f x. f 5 x) (\lambda x y z. x + y * z) 2 3} \\ & \longrightarrow \underline{(\lambda x. (\lambda x y z. x + y * z) 5 x) 2 3} \\ & \longrightarrow (\lambda x y z. x + y * z) 5 2 3 \\ & \xrightarrow{*} (5 + 2 * 3) \\ & \longrightarrow 11 \end{aligned}$$

## 2 Church Encoding

### 2.1

$$e_1 \text{ or } e_2 \stackrel{\text{def}}{=} \text{if } e_1 \text{ then true else } e_2$$

## 2.2

true **and** (false **or** true)

$\xrightarrow{\text{def}}$  **if** true **then** (false **or** true) **else** false

$\xrightarrow{\text{def}}$  true (false **or** true) false

$\xrightarrow{\text{def}}$   $(\lambda x y. x)$  (false **or** true) false

$\xrightarrow{*}$  (false **or** true)

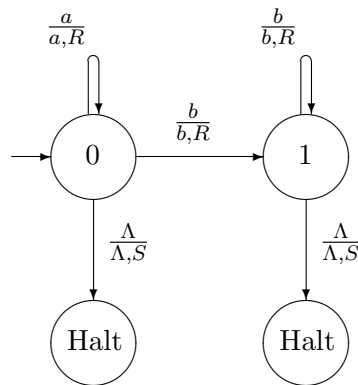
$\xrightarrow{\text{def}}$  **if** false **then** true **else** true

$\xrightarrow{\text{def}}$  false true true

$\xrightarrow{\text{def}}$   $(\lambda x y. y)$  true true

$\xrightarrow{*}$  true

## 3 A Trivial Machine

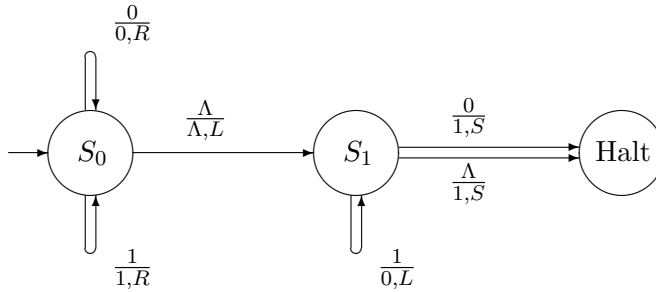


If the data is  $abb$  then the TM, starting in state 0, moves right, leaving the symbol  $a$  alone and remains in state 0. Having moved right, it finds itself in state 0 looking at a  $b$  and so it leaves the symbol unchanged, moves right and enters state 1. etc. etc. until it goes to the Halt state. We say, therefore, the string  $abb$  is accepted by this TM.

The string  $ba$  is not accepted because it gets into the state 1 with an  $a$  under the read-head. There is no transition leading away from state 1 which reads  $a$  from the tape.

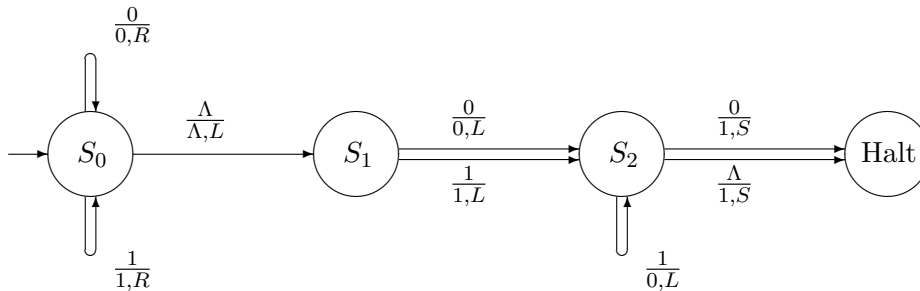
## 4 Addition of 2

Here is the machine that adds one.

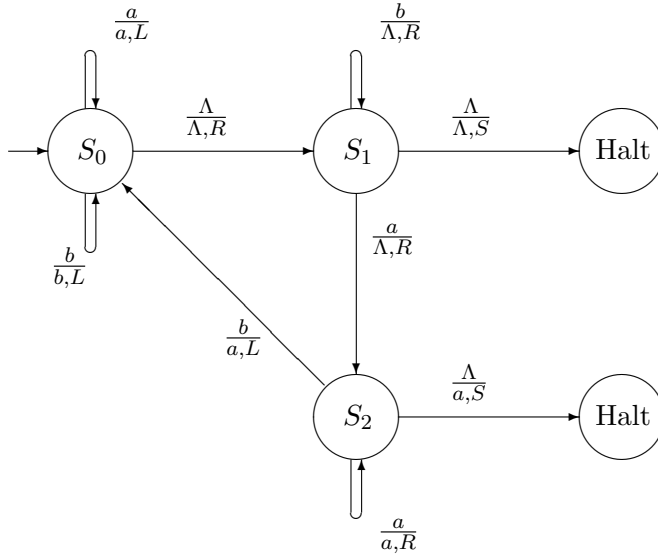


- **State  $S_0$ :** This is the state that is occupied while the TM scans right to find the least significant bit.
- **State  $S_1$ :** This state is entered when the terminating blank has been found. Then, in this state, the TM scans as far left as necessary to finish the addition. Occupancy of this state indicates that a ‘one’ is to be added to the binary number on the tape *whose last digit is at the position pointed to by the head*.
- **State Halt:** This state is only entered when the addition is complete.

An easy way of making a machine to add two to a number, would be to simply chain two copies of this add-one machine together. As we’re dealing with binary numbers, we can be a bit smarter and add one to the two’s digit instead:

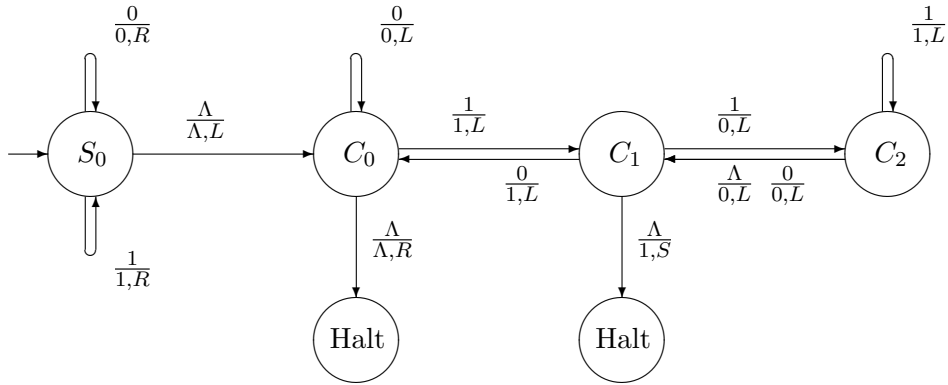


## 5 Selective Removal



- **Overview:** The TM repeatedly
  - moves to the left end of the string (state  $S_0$ )
  - deletes  $bs$  on the left end of the string (state  $S_1$ )
  - deletes the left-most  $a$
  - scans right to find a  $b$  (or blank beyond the right-hand end) and overwrites it by  $a$  (state  $S_2$ )
- **State  $S_0$ :** This is the state that is occupied while the TM scans left to find the end.
- **State  $S_1$ :** This state indicates the TM is deleting leading  $bs$  while looking for the first  $a$  on the tape, if any.
- **State  $S_2$ :** In this state the TM is carrying a single  $a$  to the right. It replaces the first  $b$  it finds by this  $a$ , or drops it on the first blank symbol if no more  $bs$  are on the tape.

## 6 Multiplication by 3



- **Overview:** The TM works from right to left along the binary number, multiplying each digit by 3, adding the “carry-in” from the previous step, and passing on the “carry-out” to the next step. Thus when  $3n + i = k + 2j$ , and  $n$  is on the tape in state  $C_i$ , write  $k$  and go to state  $C_j$ ; carry-in is  $i$  and carry-out is  $j$ .
- **State  $S_0$ :** In this state the TM scans right to find the least significant bit.
- **State  $C_i$ :** This state indicates that the carry currently is 0. The TM is to multiply the number whose rightmost digit (blank treated as 0) is under the head by 3, and add  $i$  (which is the “carry” from the previous step of the multiplication).

The important lesson here is that the state of the finite control is being used as memory.