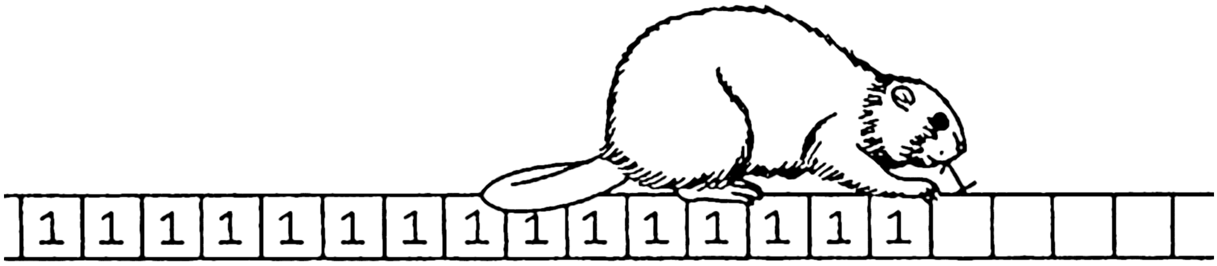
/informatik & security

Busy Beavers
Theoretical Computer Science

Dipl.-Ing. Hubert Schölnast, BSc
September 26, 2021

Table of Contents



1 An alternative definition of a Turing machine

We make some small changes in the definition of Turing machines:

- The read-write-head cannot stay on the same position. At the end of every single step it must move either to the left or to the right. It never can halt. So, the set of possible directions, which was $\{R, L, H\}$ in the standard definition, is now only $\{R, L\}$.
- Instead the machine has a special state beside the "normal" working states: the halting state Ⓗ. As soon as this special state is reached, the machine stops immediately and does nothing after that.

This new definition is fully equivalent to the standard definition, with one big difference: When the automaton has only one predefined state where it can stop, than there can't exist a difference between accepting and non-accepting states among the other working states. But for our particular purpose we don't need this differentiation.

It is easy to show that any Turing machine that satisfies the standard definition and where all states are accepting states (i.e., where there are no non-accepting states) can be transformed into a Turing machine that satisfies this new alternative definition.

For our particular purpose, we don't need any input either. The Turing machines we want to examine here all start with a completely empty tape. This is like using the empty word ε as input for a "normal" Turing machine. As a consequence, if the Turing machine is deterministic, it's behavior is completely predefined only by its alphabet and its transition function δ. And this is exactly what we want to have here. The absence of an input also means that no input alphabet is required. Let's add these new restrictions to our alternative definition:

- There is no input and therefore there also is no input alphabet. The machines always starts on an empty tape.
- The machine is deterministic.

Next, we want to have some constraints on the working alphabet. It can be proved that a Turing machine with any working alphabet, no matter how many characters it has, can be converted to another Turing machine whose alphabet has only 2 characters, and that both machines are still able to perform exactly the same tasks if only the input (which we don't need now anyway) is converted to the other alphabet. The machine with the smaller alphabet has more states and has to perform more steps, but the result is still always the same. For our purpose we want to have the smallest possible working alphabet:

- The work alphabet contains only 2 signs: ⓪ and ❶.
- The blank sign is the sign ⓪.

And because these machines are different from "normal" Turing machines, we give them a name. We call them "busy beavers". The English idiom "busy beaver" stands for someone who is very busy, very industrious, assiduous and hardworking. The Hungarian mathematician Tibor Radó (1895-1965) used this term as name for this special type of Turing machines he investigated in 1962.

We make one final restriction, which in fact is nothing else but a naming convention: We can name the working states as we want. So, let's make life easy and let's define, that the names of working states are Latin uppercase letters ($A, B, C, \cdots$). Furthermore let's define, that the initial state is always state A.

1.1 Structure of the "New" Turing Machine

B is a busy beaver, and every busy beaver is a set with exactly 2 elements:

$$B = \{Z, \delta\}$$

Z is the set of all working states.
The cardinality of Z is n: $n := |Z|$, it is a positive integer: $n \in \mathbb{N};\ n \geq 1$
If $n \leq 26$ then the names of the states are uppercase Latin characters ($A, B, C, \cdots$), among them is always state A. If $n > 26$ then among all states there are 26 states whose names are uppercase Latin characters.

δ is the transition function: $\delta: \left(Z \times \{⓪, ❶\}\right) \rightarrow \left((Z \cup \{Ⓗ\}) \times \{⓪, ❶\} \times \{L, R\}\right)$

- The busy beaver has an additional state Ⓗ, that is not a member of the working states: Ⓗ $\notin Z$. This is a state that the machine can enter. However, as soon as it reaches this state, it halts and finishes the execution. Therefore, it cannot leave this state. This is why this state is named "halting state"
- There are no accepting or non-accepting states (only non-halting working states and 1 halting state)
- The initial state is always state A.
- There is no input alphabet.
- The work alphabet is no longer variable. It is predefined. It is $\{⓪, ❶\}$ for all TMs that are conform to this alternative definition.
- The set of directions into which the head can move at the end of each step is no longer $\{L, H, R\}$ but $\{L, R\}$.
- The blank sign is no longer variable. It has a fix value. It is ⓪. This means, that at the beginning the tape (which is infinitely long in both directions) is filled with ⓪s in every single cell.

2 How many of these machines do exist?

The busy beavers only differ from each other in two parameters:

1. The number of working states $n := |Z|$
2. The exact characteristics of the transition function δ

But the second item is limited by the first one.

If $n = 1$ then there is only one working state which also is the initial state. This is state A. And this is how the signature of δ looks like in this case:

$$\delta\colon \big(\{A\} \times \{\text{⓪}, \text{❶}\}\big) \to \big(\{A, \text{Ⓗ}\} \times \{\text{⓪}, \text{❶}\} \times \{L, R\}\big)$$

The machine is deterministic by definition. This means, the function δ has only two elements:

$$\delta \ = \begin{Bmatrix} (A, \text{⓪}) \to x, \\ (A, \text{❶}) \to y \end{Bmatrix}$$

The right sides of the function's elements (here written as x and y) are elements of the Cartesian product of three sets where each of them contains exactly 2 elements:

$$x, y \in \{A, \text{Ⓗ}\} \times \{\text{⓪}, \text{❶}\} \times \{L, R\}$$

So, both, x and y, are sequences of length 3, and there are exactly 8 possible sequences:

$$\{A, \text{Ⓗ}\} \times \{\text{⓪}, \text{❶}\} \times \{L, R\} =$$

$$= \{(A, \text{⓪}, L), (A, \text{⓪}, R), (A, \text{❶}, L), (A, \text{❶}, R), (\text{Ⓗ}, \text{⓪}, L), (\text{Ⓗ}, \text{⓪}, R), (\text{Ⓗ}, \text{❶}, L), (\text{Ⓗ}, \text{❶}, R), \}$$

In fact it doesn't really matter where the head is positioned after the automaton has reached state Ⓗ, so there is not really a difference between $(\text{Ⓗ}, \text{⓪}, L)$ and $(\text{Ⓗ}, \text{⓪}, R)$ and between $(\text{Ⓗ}, \text{❶}, L)$ and $(\text{Ⓗ}, \text{❶}, R)$, but the functions are still different, and so still have 8 elements in the Cartesian product.

All in all we have this situation:

In the element $(A, \text{⓪}) \to x$ there are 8 possible values for x and in $(A, \text{❶}) \to y$ there are 8 possible values for y which are independent from the possibilities for x.

And this means:

- There are exactly 64 ($8 \cdot 8 = 64$) different busy beavers for $n = 1$.

Before we examine what these machines can do, let's look at how many machines there are for other sizes of Z:

$n = 2$ means $Z = \{A, B\}$ and therefore:

$$\delta\colon \big(\{A, B\} \times \{\text{⓪}, \text{❶}\}\big) \to \big(\{A, B, \text{Ⓗ}\} \times \{\text{⓪}, \text{❶}\} \times \{L, R\}\big)$$

The cartesian product of the two sets on the left side of the arrow has $2 \cdot 2 = 4$ elements, and this means that δ has 4 elements:

$$\delta = \begin{Bmatrix} (A, \textcircled{0}) \to x_1, \\ (A, \text{❶}) \to x_2, \\ (B, \textcircled{0}) \to x_3, \\ (B, \text{❶}) \to x_4 \end{Bmatrix}; \quad |\delta| = 4$$

Obviously this number is exactly twice as large as n:

$$|\delta| = |Z| \cdot |\{\textcircled{0}, \text{❶}\}|$$
$$|\delta| = n \cdot 2 = 2 \cdot 2 = 4$$

The cartesian product on the right side has 12 elements:

$$|\{A, B, \textcircled{H}\}| = 3 \;= n + 1$$
$$|\{\textcircled{0}, \text{❶}\}| \;= 2$$
$$|\{L, R\}| \;= 2$$
$$3 \cdot 2 \cdot 2 \;= 12$$

So, the total number of busy beavers with 2 working states is

$$12 \cdot 12 \cdot 12 \cdot 12 = 12^4 = 20736$$

It sound be clear now, that for a given n the number of elements in the function δ is:

$$|\delta| = n \cdot 2$$

And the codomain has this many elements:

$$\left|(Z \cup \{\textcircled{H}\}) \times \{\textcircled{0}, \text{❶}\} \times \{L, R\}\right| = (n+1) \cdot 2 \cdot 2 = (n+1) \cdot 4 = 4n + 4$$

And this means, that there are in total $(4n + 4)^{2n}$ busy beavers with n working states.

number of states	number of busy beavers
1	64
2	20,736
3	16,777,216
4	25,600,000,000
5	63,403,380,965,376
6	232,218,265,089,212,416
7	1,180,591,620,717,411,303,424
8	7,958,661,109,946,400,884,391,936
9	68,719,476,736,000,000,000,000,000,000
10	739,696,442,014,594,807,059,393,047,166,976
11	9,711,967,541,295,580,042,210,555,933,809,967,104
12	152,784,834,199,652,075,368,661,148,843,397,208,866,816
13	2,837,191,840,326,756,799,991,596,824,571,432,237,117,997,056
14	61,409,422,144,648,154,972,160,000,000,000,000,000,000,000,000,000
15	1,532,495,540,865,888,858,358,347,027,150,309,183,618,739,122,183,602,176

3 How many ❶s can a busy beaver write on the tape?

3.1 Busy beavers with 1 working state

This is the central question that Tibor Radó has investigated. Let's start with busy beavers with only 1 state. We can quickly find 4 different types of busy beavers:

Type A, shown in this example:

$$\delta = \begin{Bmatrix} (A, \text{⓪}) \to (A, \text{❶}, R), \\ (A, \text{❶}) \to (A, \text{⓪}, R) \end{Bmatrix}$$

Busy beavers of this type can never enter state Ⓗ. The particular busy beaver shown here always stays in state A and writes an infinitely long series of ❶s on the tape. But this is cheating. Radó wants that the beavers halt, and only then the number of ❶s shall be counted. So, busy beavers of this Type are disqualified because they run forever.

Type B, shown in this example:

$$\delta = \begin{Bmatrix} (A, \text{⓪}) \to (A, \text{❶}, R), \\ (A, \text{❶}) \to (\text{Ⓗ}, \text{⓪}, R) \end{Bmatrix}$$

Beavers of this type have the possibility to enter the halting state Ⓗ, but they still never will do it, because the combination of state and sign on the tape, that will bring the automaton into state Ⓗ can never be reached. This particular machines starts in state A and finds an ⓪ on the tape. It stays in state A replaces the sign on the tape by a ❶ and then moves to the right. There it finds another ⓪, so it does the same. It never walks back and therefore never will see any of the ❶s it has written before. Busy beavers of this Type are also disqualified because they also run forever.

Type C, shown on this example:

$$\delta = \begin{Bmatrix} (A, \text{⓪}) \to (\text{Ⓗ}, \text{❶}, R), \\ (A, \text{❶}) \to (A, \text{⓪}, R) \end{Bmatrix}$$

This particular busy beaver write exactly one ❶ and then halts. It's behavior is easy to predict, so it's easy to tell that it will halt for sure and easy to tell how many ❶s it will write on the tape.

Type D (In the set of busy beavers with $n = 1$ there is no example of this type): Busy beavers of this type contain at least one transition in δ that might bring the automaton into the halting state Ⓗ, but just by analyzing the transition function δ it is very hard to tell if they will halt, and if they do, how many ❶s they will write on the tape. The best way to find it out is to simulate this busy beaver and watch what it does.

If you play around with δ for busy beavers with $n = 1$, you will quickly see, that those machines either run forever, because they never can reach state Ⓗ, or they reach state Ⓗ in the very first step. There can't be a busy beaver with only 1 working state, that halts after 2 or more steps. And 50% of those who halt, do this by writing a ⓪ and the other half write exactly one ❶. So, the maximum number of ❶s a busy beaver with 1 state can write, is: 1.

3.2 Best busy beaver with 2 working states

There are always more than one busy beavers that can write the maximum number of ❶s, and here is a busy beaver with 2 states, that can write 4 ❶s:

$$\delta = \begin{Bmatrix} (A,⓪) \rightarrow (B,❶,R), \\ (A,❶) \rightarrow (B,❶,L), \\ (B,⓪) \rightarrow (A,❶,L), \\ (B,❶) \rightarrow (Ⓗ,❶,R) \end{Bmatrix}$$

It writes 4 ❶s and needs 6 steps until it halts. You can find this out just with pen and paper if you want.

3.3 Best busy beaver with 3 working states

Here is one of the winners in the class $n = 3$:

$$\delta = \begin{Bmatrix} (A,⓪) \rightarrow (B,❶,R), \\ (A,❶) \rightarrow (Ⓗ,❶,R), \\ (B,⓪) \rightarrow (C,⓪,R), \\ (B,❶) \rightarrow (B,❶,R), \\ (C,⓪) \rightarrow (C,❶,L), \\ (C,❶) \rightarrow (A,❶,L) \end{Bmatrix}$$

This busy beaver performs 14 steps and writes 6 ❶s.

The best busy beavers with 1, 2 and 3 states have been found by Radó in the years from 1962 to 1965.

3.4 Best busy beaver with 4 working states

Here is the winner in the class $n = 4$:

$$\delta = \begin{Bmatrix} (A,⓪) \rightarrow (B,❶,R), \\ (A,❶) \rightarrow (B,❶,L), \\ (B,⓪) \rightarrow (A,❶,L), \\ (B,❶) \rightarrow (C,⓪,L), \\ (C,⓪) \rightarrow (Ⓗ,❶,R), \\ (C,❶) \rightarrow (D,❶,L), \\ (D,⓪) \rightarrow (D,❶,R), \\ (D,❶) \rightarrow (A,⓪,R) \end{Bmatrix}$$

It needs 107 steps to write 13 ❶s and was found in 1972.

3.5 Best busy beaver with 5 working states

There are 63.4 trillion ($63.4 \cdot 10^{12}$) different busy beavers in this class, and among them there is a considerable number of type D.

This here belonged for a long time to this class too, but now it is clear: It halts after 47,176,870 steps, and then there are 4098 ❶s on the tape (with 8191 ⓪s interspersed). It was found in 1989.

$$\delta = \begin{Bmatrix} (A,\text{⓪}) \rightarrow (B,\text{❶},R), \\ (A,\text{❶}) \rightarrow (C,\text{❶},L), \\ (B,\text{⓪}) \rightarrow (C,\text{❶},R), \\ (B,\text{❶}) \rightarrow (B,\text{❶},R), \\ (C,\text{⓪}) \rightarrow (D,\text{❶},R), \\ (C,\text{❶}) \rightarrow (E,\text{⓪},L), \\ (D,\text{⓪}) \rightarrow (A,\text{❶},L), \\ (D,\text{❶}) \rightarrow (D,\text{❶},L), \\ (E,\text{⓪}) \rightarrow (\text{Ⓗ},\text{❶},R), \\ (E,\text{❶}) \rightarrow (A,\text{⓪},L) \end{Bmatrix}$$

Until now no other busy beaver with 5 states was found that halts and then has more than 4098 ❶s written on the tape. But there are still 40 busy beavers with 5 states, where it is unclear if they will run forever or if they will halt, and if they halt, it is unclear, how many ❶s they will write on the tape. These 40 busy beavers are still running for more than 30 years, and it still is unclear if they will halt some day or not.

3.6 Best busy beaver with 6 working states

$$\delta = \begin{Bmatrix} (A,\text{⓪}) \rightarrow (B,\text{❶},R), \\ (A,\text{❶}) \rightarrow (E,\text{❶},L), \\ (B,\text{⓪}) \rightarrow (C,\text{❶},R), \\ (B,\text{❶}) \rightarrow (F,\text{❶},R), \\ (C,\text{⓪}) \rightarrow (D,\text{❶},L), \\ (C,\text{❶}) \rightarrow (B,\text{⓪},R), \\ (D,\text{⓪}) \rightarrow (E,\text{❶},R), \\ (D,\text{❶}) \rightarrow (C,\text{⓪},L), \\ (E,\text{⓪}) \rightarrow (A,\text{❶},L), \\ (E,\text{❶}) \rightarrow (D,\text{⓪},R), \\ (F,\text{⓪}) \rightarrow (\text{Ⓗ},\text{❶},L), \\ (F,\text{❶}) \rightarrow (C,\text{❶},R) \end{Bmatrix}$$

This busy beaver was found in 2010. It could be proved by using several techniques (tape compression, macro machines, automatic proof systems etc.) that this machine will halt. It runs through $7.412 \cdot 10^{36534}$ steps. This is a number with 36534 decimal digits. Written out in full, this number would be 12 pages long in this document. This busy beaver writes $3.515 \cdot 10^{18267}$ ❶s. This number would fill 6 pages of this document.

3.7 Best busy beaver with 7 working states

Until now no concrete candidate has been found, but by mathematical methods, similar to that, used for the 6-state busy beavers, it could be proved, that the number of ❶s written by the best 7-state busy beaver must be greater than

$$10^{10^{10^{10^{18705353}}}}$$

Let a be this number, and let b be the number of digits of a. Then we can define c as the number of digits of b and d as the number of digits of c. Then d has $18{,}705{,}353$ decimal digits and would fill more than 6200 pages of this document.