Busy Beaver Problem

By Jonathan Ho

Introduction to Computability Theory and Recursive Functions

- Computability Theory originated from early logicians in the late 1930s
  - Church, Godel, Kleene, Post, Rosser, Turing
- A function is **computable** if there exists an algorithm that can do the job of the function
  - Church-Turing Thesis: a function is **computable** if it can be calculated on a Turing Machine with unlimited time and space
  - Ex: Addition Function - $A(n_1, n_2) = n_1+n_2$ | $A(0, 0) = 0$; $A(-3, 9) = 6$; $A(0.3, 0.7) = 1.0$
- A function is **recursive** if it maps from natural numbers to natural numbers

What is Recursive Unsolvability?

- A set or problem is **unsolvable** (undecidable) if its characteristic function is not computable.
- Basically, if we have a set *W* and a subset *A*, we decide whether each element in *W* is in *A* with some algorithm. If there is such a procedure for A, it is said to be decidable. If there is no such algorithm, then the set is unsolvable/undecidable.

Examples of decision problems that are recursively unsolvable

- Hilbert's 10th problem
- Halting Problem
- Gödel numbering

History behind Busy Beaver Problem

- First introduced by Tibor Rado in his 1961 paper, *On Non-Computable Functions*
- His goal was to teach beginners about Turing Machines and non-computable functions
- Thus, he created the Busy Beaver problem as a simple way to explore these concepts
- In fact, no enumeration or diagonalization is needed to prove the function is undecidable
- Instead, Rado used that fact that a non-empty finite set of integers must have a largest element

What is the Busy Beaver Problem?

- Uses an infinite-tape Turing Machine
- As a simplification, the term **card** is used instead of state. Each card contains 3 values, for each possible ***scanned*** values: the ***overprint*** value, ***shift*** value, and ***call*** card value.
- Starting Card: 0; Halting Card: -1; 0 is a *left* shift and 1 is a *right* shift
- Question: Using a Turing Machine with *n* cards, what is the **maximum** number of nonzero characters that can be printed on the tape when it **halts** starting from an all-zero tape?

Card	1
0	100
1	11-1

This is an example card. When the machine is on this card and a 0 is scanned, a 1 is printed on the tape, the tape head shifts one spot to the left, and we switch to Card 0. When the machine is on this card and a 1 is scanned, a 1 is printed on the tape, the tape head shifts one spot to the right, and we reach the halting card.

Functions derived from Busy Beaver Machines

Define the following 2 functions for some machine *M*

- **s(M)**: the number of shifts *M* takes before halting
- **σ(M)**: the number of 1's on the tape when *M* halts

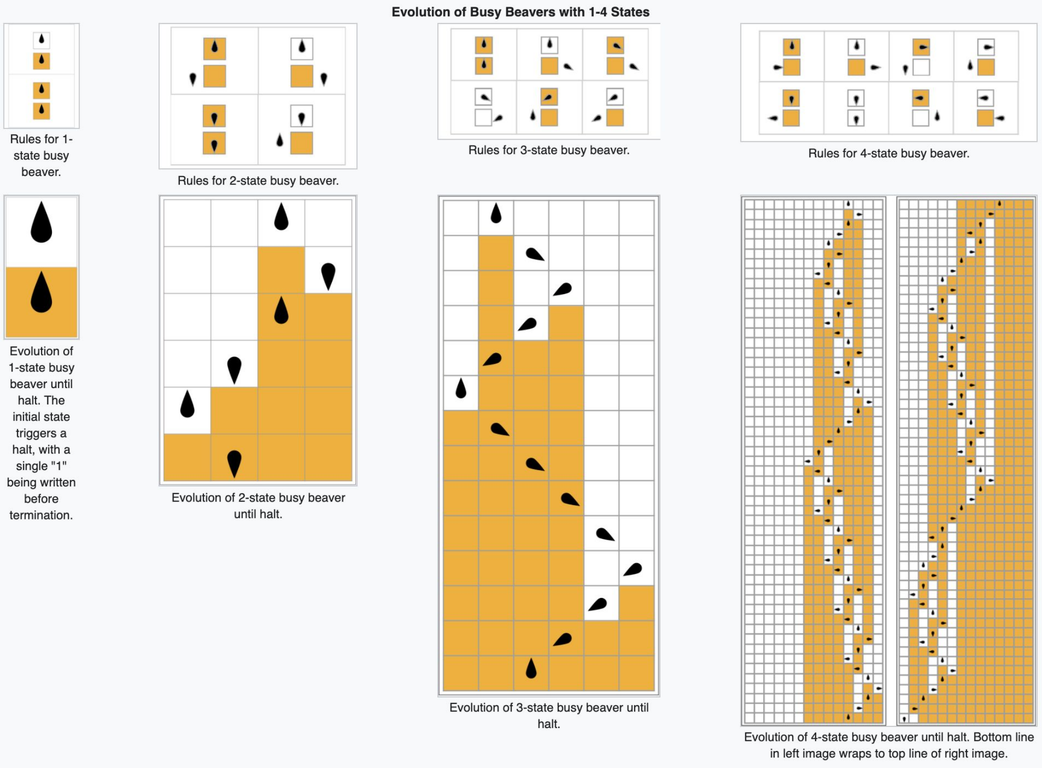
The following functions all grow at an exponential rate as *n* increases:

- **N(n)**: the number of possible *n*-card Turing Machines
- **S(n):** max{ s(M) | M is some *n*-card Turing Machine } (**Maximum Shift Function**)
- **Σ(n):** max{ σ(M) | M is some *n*-card Turing Machine } (**Busy Beaver Function**)

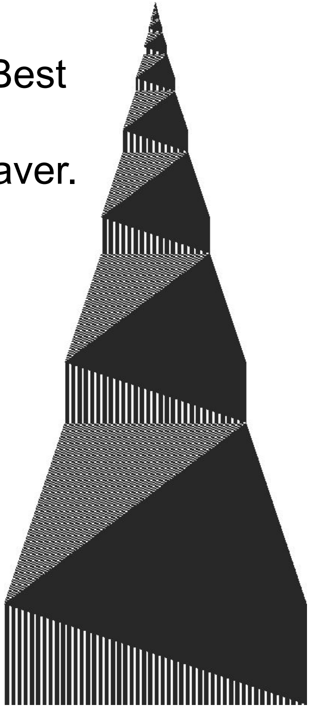
Current Values for the Functions

n	$N(n) = (4n+4)^{2n}$	$\Sigma(n)$	$S(n)$
1	64	1	1
2	20,736	4	6
3	1.7×10^{7}	6	21
4	2.6×10^{10}	13	107
5	6.3×10^{13}	> 4,098 ?	> 47,176,870 ?
6	2.3×10^{17}	$> 3.5 \times 10^{18,267}$?	$> 7.4 \times 10^{36,534}$?

Visual Representations of the Best Machines



Current Best
5-state
Busy Beaver.



What if we extend this to m symbols?

- Then the following functions all grow at an exponential rate:
  - **N(n, m)** = $[2m(n+1)]^{mn}$
  - **S(n, m)**
  - **Σ(n, m)**

Values of S(*n*, *m*)						
m \ *n*	2-state	3-state	4-state	5-state	6-state	7-state
2-symbol	6	21	107	47 176 870 ?	$> 7.4 \times 10^{36\,534}$	$> 10^{10^{10^{10^{18\,705\,353}}}}$
3-symbol	38	≥ 119 112 334 170 342 540	$> 1.0 \times 10^{14\,072}$	?	?	?
4-symbol	≥ 3 932 964	$> 5.2 \times 10^{13\,036}$	?	?	?	?
5-symbol	$> 1.9 \times 10^{704}$	?	?	?	?	?
6-symbol	$> 2.4 \times 10^{9866}$	?	?	?	?	?
Values of Σ(*n*, *m*)						
m \ *n*	2-state	3-state	4-state	5-state	6-state	7-state
2-symbol	4	6	13	4098 ?	$> 3.5 \times 10^{18\,267}$	$> 10^{10^{10^{10^{18\,705\,353}}}}$
3-symbol	9	≥ 374 676 383	$> 1.3 \times 10^{7036}$	?	?	?
4-symbol	≥ 2050	$> 3.7 \times 10^{6518}$	?	?	?	?
5-symbol	$> 1.7 \times 10^{352}$	?	?	?	?	?
6-symbol	$> 1.9 \times 10^{4933}$	?	?	?	?	?

Frontiers of the Busy Beaver Problem

- Computing Σ(n, m) and S(n, m) for further values of n and m
- Relationship between smaller busy beaver instances and larger instances
- Uniqueness of the best busy beavers and maximizing functions
- Behavior on non-zero inputs
- Busy beaver machine represented as directed graphs where cards are vertices and transitions are edges
- Behavior when busy beaver machine has access to an oracle

References and Further Reading

Aaronson, Scott. The Busy Beaver Frontier. https://www.scottaaronson.com/papers/bb.pdf

Computerphile. Busy Beaver Turing Machines. https://www.youtube.com/watch?v=CE8UhcyJS0I

Dean, Walter. Recursive Functions. https://plato.stanford.edu/entries/recursive-functions/

Jones, James. Recursive Undecidability - An Exposition. https://www.jstor.org/stable/2319560?seq=1

Michel, Pascal. The Busy Beaver Competition - Current Records. http://www.logique.jussieu.fr/~michel/bbc.html

Michel, Pascal. Table and Timeline of Historical Results. http://www.logique.jussieu.fr/~michel/ha.html

Rado, Tibor. On Non-Computable Functions. http://computation4cognitivescientists.weebly.com/uploads/6/2/8/3/6283774/rado-on_non-computable_functions.pdf

Sakharov, Alex. Recursively Undecidable from MathWorld. https://mathworld.wolfram.com/RecursivelyUndecidable.html

Weisstein, Eric. Busy Beaver from MathWorld. https://mathworld.wolfram.com/BusyBeaver.html

Zenil, Hector. Busy Beaver Wolfram Demonstration Project. https://demonstrations.wolfram.com/BusyBeaver/