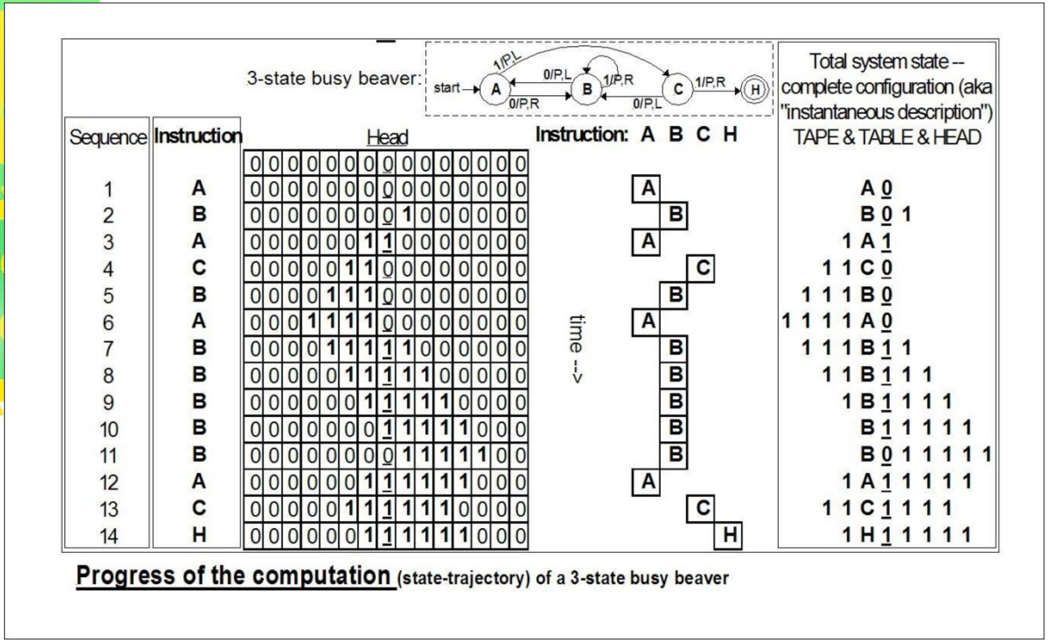
The Busy Beaver Frontier



3-state busy beaver:

$$\begin{tikzcd} \text{start} \arrow[r] & A \arrow[r, "0/P{,}R"'] \arrow[rr, bend left, "1/P{,}L"] & B \arrow[l, bend right, "0/P{,}L"'] \arrow[loop above, "1/P{,}R"] & C \arrow[l, "0/P{,}L"] \arrow[r, "1/P{,}R"] & H \end{tikzcd}$$

Sequence	Instruction	Head	Instruction: A B C H	Total system state -- complete configuration (aka "instantaneous description") TAPE & TABLE & HEAD
		0 0 0 0 0 0 0 $\underline{0}$ 0 0 0 0 0 0 0		
1	A	0 0 0 0 0 0 0 $\underline{0}$ 0 0 0 0 0 0 0	A	A $\underline{0}$
2	B	0 0 0 0 0 0 0 $\underline{0}$ 1 0 0 0 0 0 0	B	B $\underline{0}$ 1
3	A	0 0 0 0 0 0 1 $\underline{1}$ 0 0 0 0 0 0 0	A	1 A $\underline{1}$
4	C	0 0 0 0 0 1 1 $\underline{0}$ 0 0 0 0 0 0 0	C	1 1 C $\underline{0}$
5	B	0 0 0 0 1 1 1 $\underline{0}$ 0 0 0 0 0 0 0	B	1 1 1 B $\underline{0}$
6	A	0 0 0 1 1 1 1 $\underline{0}$ 0 0 0 0 0 0 0	A	1 1 1 1 A $\underline{0}$
7	B	0 0 0 0 1 1 1 $\underline{1}$ 1 0 0 0 0 0 0	B	1 1 1 B $\underline{1}$ 1
8	B	0 0 0 0 0 1 1 $\underline{1}$ 1 1 0 0 0 0 0	B	1 1 B $\underline{1}$ 1 1
9	B	0 0 0 0 0 0 1 $\underline{1}$ 1 1 1 0 0 0 0	B	1 B $\underline{1}$ 1 1 1
10	B	0 0 0 0 0 0 0 $\underline{1}$ 1 1 1 1 0 0 0	B	B $\underline{1}$ 1 1 1 1
11	B	0 0 0 0 0 0 0 $\underline{0}$ 1 1 1 1 1 0 0	B	B $\underline{0}$ 1 1 1 1 1
12	A	0 0 0 0 0 0 1 $\underline{1}$ 1 1 1 1 0 0 0	A	1 A $\underline{1}$ 1 1 1 1
13	C	0 0 0 0 0 1 1 $\underline{1}$ 1 1 1 0 0 0 0	C	1 1 C $\underline{1}$ 1 1 1
14	H	0 0 0 0 0 0 1 $\underline{1}$ 1 1 1 1 0 0 0	H	1 H $\underline{1}$ 1 1 1 1

time -->

Progress of the computation (state-trajectory) of a 3-state busy beaver

Scott Aaronson (University of Texas at Austin)

UC Davis, October 15, 2020

You have 15 seconds. What's the biggest integer you can name?

99999999999999999999

$9^{9^{9^{9^{9^{9^{9}}}}}}$:= ${}^{7}9$ ${}^{{}^{{}^{{}^{{}^{{}^{9}9}9}9}9}9}9$

Ackermann(1000)

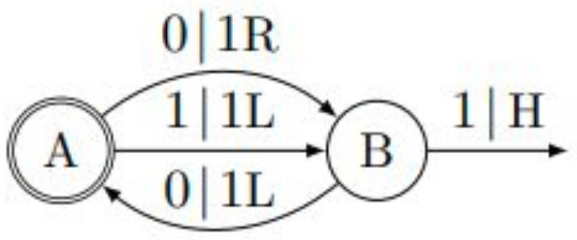
Graham's Number G

The largest integer that be named in at most 1000 words **+1**

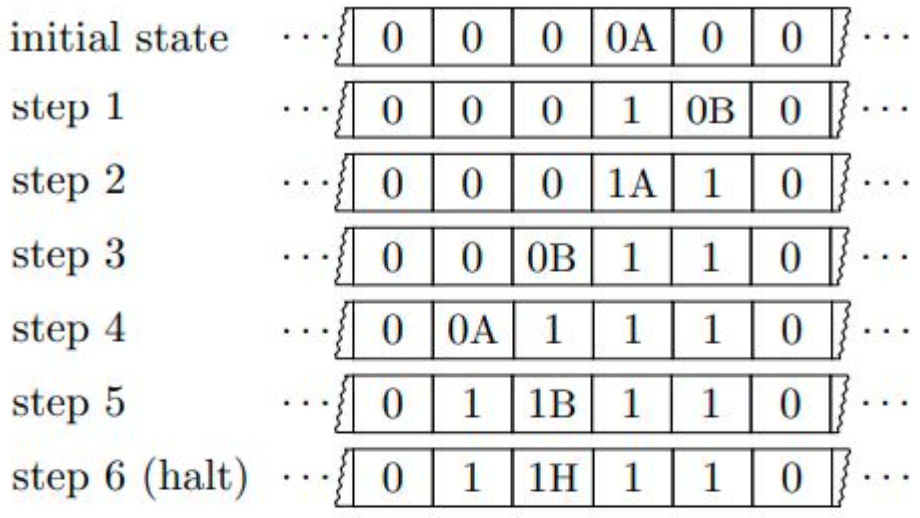
The largest integer that be named using a 1000-bit computer program

Busy Beaver Function (Radó 1962)

Turing Machines
(1 tape, 2 symbols)



$$\begin{tikzcd}
A \arrow[r, bend left, "0|1R"] \arrow[r, "1|1L"] & B \arrow[l, bend left, "0|1L"] \arrow[r, "1|H"] & {}
\end{tikzcd}$$



initial state	···	0	0	0	0A	0	0	···
step 1	···	0	0	0	1	0B	0	···
step 2	···	0	0	0	1A	1	0	···
step 3	···	0	0	0B	1	1	0	···
step 4	···	0	0A	1	1	1	0	···
step 5	···	0	1	1B	1	1	0	···
step 6 (halt)	···	0	1	1H	1	1	0	···

Given a TM M, let s(M) be the number of steps M runs for on a blank tape. Then BB(n) is the max of s(M), over all n-state TMs M with s(M)<∞. "Busy Beavers" are M's that achieve the maximum.

Examples: BB(1)=1. The 2-state TM above shows that BB(2)≥6. In fact BB(2)=6.

BB grows uncomputably quickly!

Theorem: Computing any upper bound $f(n) \geq BB(n)$ is equivalent to solving the halting problem.

Proof: For BB≤HALT, just take the max over n-state halting TMs. For HALT≤f, run an n-state TM for f(n) steps. If it hasn't halted by then, it never will.

Theorem: For every computable function f, there exists an n_f such that $BB(n) > f(n)$ for all $n \geq n_f$.

Proof: For any n, we can design a TM with $c_f + O(\log n)$ states that computes f(n) and then stalls for (say) $f(n)^2$ steps.

BB eludes formal systems

Theorem: Let F be a reasonable formal system (like PA or ZFC). Then there exists a constant n_F such that F can't prove the value of BB(n) for any $n \geq n_F$.

Proof: Suppose not. Then we could compute BB(n) for any n, by enumerating over all possible proofs.

Did we just reprove a version of Gödel's Incompleteness Theorem? Yes we did!

Proof #2: Let M_F be an n-state TM that enumerates the theorems of F, halting iff it finds a contradiction. If F proved the value of BB(n), it would prove that M_F ran forever, and hence F's own consistency

Think about that…

For every consistent large cardinal axiom, its consistency is implied by some statement of the form "BB(n)=k"

Is every Busy Beaver number determined by **some** consistent large cardinal axiom? Maybe, but if so, there's no computable way to **find** those axioms!

More broadly, the first 1000 BB numbers encode a large portion of all interesting mathematical truth!
BB(27): Goldbach Conjecture / BB(744): Riemann Hypothesis…

"The BB Argument for Arithmetical Platonism"?

Beyond Busy Beaver?

Theorem: Let $BB_1(n)$ be the BB function for TM’s with oracles for ordinary BB. Then BB_1 grows faster than any function computable with a BB oracle.

Proof: The uncomputability of BB relativizes!

In general, for any ordinal α, let $BB_{\alpha+1}(n)$ be BB for TM’s with oracles for BB_α. Or if β is defined as $\lim_{n\to\infty}\beta(n)$, then let $BB_\beta(n):=BB_{\beta(n)}(n)$.

How much further can we go, without our numbers depending on the intended model of set theory?

Intermediate growth rates

Theorem: There’s a function g:N⍰N that dominates every computable function f, yet such that BB and HALT are *still* uncomputable given an oracle for g

Proof: Let $f_1,f_2,\ldots$:N⍰N be an enumeration of computable functions. We set

$$g(n) := \max_{i \leq w(n)} f_i(n),$$

for some nondecreasing w that increases without bound—thereby ensuring that g dominates every f_i. For the other property, only increment w (i.e., set w(n+1)=w(n)+1) after another candidate reduction from HALT to g has been “killed off”

Concrete Values

n	BB(n)	**Reference**
1	1	Trivial
2	6	Lin 1963
3	21	Lin 1963
4	107	Brady 1983
5	$\geq$ 47,176,870	Marxen and Buntrock 1990
6	$> 7.4 \times 10^{36{,}534}$	Kropitz 2010
7	$> 10^{2\times 10^{10^{10^{18{,}705{,}352}}}}$	"Wythagoras" 2014

BB(18) >> Graham's number >> Ackermann(18)

What's the least n with BB(n)>Ackermann(n)?

What does the 5-state champ *do*?

Consider the "Collatz-like" map g:N→N ∪ {⊥}:

$$g(x) := \begin{cases} \frac{5x+18}{3} & \text{if } x \equiv 0 \pmod 3 \\ \frac{5x+22}{3} & \text{if } x \equiv 1 \pmod 3 \\ \perp & \text{if } x \equiv 2 \pmod 3 \end{cases}$$

Starting from 0, does iterating g ever reach ⊥?

$$0 \to 6 \to 16 \to 34 \to 64 \to 114 \to 196 \to 334 \to 564$$
$$\to 946 \to 1584 \to 2646 \to 4416 \to 7366 \to 12284 \to \perp.$$

The current 5-state BB champion verifies this fact.

How many BB values are knowable?

Theorem (O’Rear, building on A.-Yedidia): There’s a 748-state TM that halts iff there’s an inconsistency in ZFC. Thus, if ZFC is consistent, then it can’t prove the value of BB(748)

To get from ~1,000,000 down to 748 took a lot of optimizations!

Is the value of BB(20) provable in ZFC? Will we ever know BB(6)?

Is there a gap between the first BB(n) value that’s **unprovable in ZFC**, and the first BB(n’) value (n’≥n) that **implies Con(ZFC)**?

BB(n) vs. BB(n+1)

"Obvious fact": $BB(n+1) > 2^{BB(n)}$ for all large enough n

This remains open!! Incredibly, the best we know (from Bruce Smith) is **$BB(n+1) \geq BB(n)+3$** for all n

Theorem (Ben-Amram and Petersen 2002): For every computable function f, there exists a c_f such that $BB(n+8\lceil n/\log n\rceil+c_f) > f(BB(n))$ for all n.

Proof Idea: "Introspective encoding." For every n-state TM M, there's an $n+O(n/\log(n))$-state TM that writes a description of M onto its tape

Chaitin's Problem

If you knew BB(n), how many bits would someone need to tell you to let you compute BB(n+1)?

Theorem (Chaitin): Let L be a programming language where no valid program is a proper prefix of another. Let BB_L be BB for L-programs. Then $BB_L(n+1)$ is computable from $BB_L(n)$ plus O(log n) bits.

Proof uses the famous Chaitin's constant:

$$\Omega_L := \sum_{L\text{-programs } P \text{ that halt}} 2^{-|P|}$$

Theorem (A.): BB(n+1) is computable from BB(n) plus O(n) bits *(beats the trivial O(n log n))*

Lazy Beavers

Define the nth **Lazy Beaver** number, LB(n), to be the least t such that there's no n-state Turing machine that runs for exactly t steps

n	1	2	3	4	5	6
$\mathrm{LB}(n)$	2	7	22	72	427	33,851

Unlike BB, LB is computable! Furthermore, $\mathrm{LB}(n) \le (4n+1)^{2n}+1$ by a counting argument

Theorem (A.-Smith, in preparation): LB(n) grows like $n^{\Omega(n)}$, and requires $n^{\Omega(n)}$ time to compute.

Beeping Busy Beavers (A.-Friedman)

A "beeping Turing machine" never halts, but has a state that emits a "beep"

Given a TM M, let b(M) be the last time step where M beeps on an all-0 input, or ∞ if there isn't one. Then let BBB(n) be the max of b(M), among all n-state machines M for which b(M)<∞

n	1	2	3	4
BB (n)	1	6	21	107
BBB(n)	1	6	≥ 55	$\geq 66{,}349$

Theorem: BBB(n) grows uncomputably quickly *even given an oracle for BB(n) (indeed, like BB_1)*

Curious Questions

For which n’s is BB(n) odd? Prime? A perfect square? Are there infinitely many such n’s? Given n, is it decidable whether BB(n) has these properties?

Does every Busy Beaver halt on *all* finite inputs?

Does every Busy Beaver have a strongly connected graph?

For n≥3, is there an essentially unique n-state Busy Beaver?