Determination of the fifth Busy Beaver value

Tristan Stérin[1] and The bbchallenge Collaboration[2]

[1]prgm.dev, Paris, France, tristan@prgm.dev
[2]bbchallenge.org, bbchallenge@bbchallenge.org

Introduced by Tibor Radó in 1962 as part of the *Busy Beaver game* [7], the Busy Beaver value $S(n)$ is the maximum number of steps that a halting n-state 2-symbol Turing machine can do from the all-0 tape before halting. The function S is noncomputable as it would otherwise allow solving the halting problem (in its input-less variant): run an n-state machine for $S(n) + 1$ steps and, if it has not halted by then, you know it will never halt.

The Busy Beaver function S is relevant in the context of molecular computing, as it enables the construction of (extremely) large structures, or long-running computations, from small (thus potentially experimentally realisable) molecular programs. To date, it has mainly been used in various models of tile assembly [8, 3, 4] but also in polymer reaction networks [5].

Since it is noncomputable, there is no algorithm that computes *all* values of S, but one can still try to determine *some* of them. Prior to this work, only the first four values had been proved: $S(1) = 1$, $S(2) = 6$ [7], $S(3) = 21$ [6], $S(4) = 107$ [2]. In 1989, Marxen and Buntrock proved that $S(5) \geq 47{,}176{,}870$ by finding a 5-state Turing machine achieving this step count.[1] In 2020, after thirty years without improvements, Aaronson conjectured that there was no better 5-state machine, i.e. conjecturing $S(5) = 47{,}176{,}870$ [1].

Our main result is to prove this conjecture, using the Coq / Rocq proof assistant [9]:

Theorem (`Lemma BB5_value`)**.** $S(5) = 47{,}176{,}870$.

The proof enumerates 181,385,789 Turing machines and proves each of them halting or non-halting using novel automated proof strategies (or individual proofs for 13 particularly hard cases called "Sporadic Machines") developed by our massively collaborative online research community, `bbchallenge.org`.

We will present the outline of the proof, including (i) the enumeration of Turing machines; (ii) the main new ideas behind automated proof strategies; (iii) the difficulties faced with Sporadic Machines. We will also discuss emerging trends in massively collaborative mathematics.

[1] S. Aaronson. The Busy Beaver Frontier. *SIGACT News*, 51(3):32–54, Sept. 2020.

[2] A. H. Brady. The determination of the value of rado's noncomputable function $\Sigma(k)$ for four-state turing machines. *Mathematics of Computation*, 40(162):647–665, 1983.

[3] D. Caballero, T. Gomez, R. Schweller, and T. Wylie. Verification and computation in restricted tile automata. *Natural Computing*, 23(2):387–405, Jun 2024.

[4] S. Cannon, E. D. Demaine, M. L. Demaine, S. Eisenstat, D. Furcy, M. J. Patitz, R. Schweller, S. M. Summers, and A. Winslow. On the effects of hierarchical self-assembly for reducing program-size complexity. *Theoretical Computer Science*, 894:50–78, 2021. Building Bridges – Honoring Nataša Jonoska on the Occasion of Her 60th Birthday.

[5] R. F. Johnson and E. Winfree. Verifying polymer reaction networks using bisimulation. *Theoretical Computer Science*, 843:84–114, 2020.

[6] S. Lin. *Computer studies of Turing machine problems*. PhD thesis, Ohio State University, Graduate School, 1963.

[7] T. Radó. On non-computable functions. *Bell System Technical Journal*, 41(3):877–884, 1962.

[8] P. W. K. Rothemund and E. Winfree. The program-size complexity of self-assembled squares (extended abstract). STOC '00, page 459–468, New York, NY, USA, 2000. Association for Computing Machinery.

[9] T. C. D. Team. The Coq Proof Assistant v8.20, Dec. 2024.

[1]`https://bbchallenge.org/1RB1LC_1RC1RB_1RD0LE_1LA1LD_1RZ0LA`