CDM

Wild Computation

Klaus Sutner

Carnegie Mellon University

Fall 2025

1 Hunting Busy Beavers

2 Towards a Proof

3 Goodstein Sequences

BB State of the Art

n	$\mathsf{BB}_{\mathrm{H}}(n)$	$\mathsf{BB}_{\mathrm{W}}(n)$
1	1	1
2	6	4
3	21	6
4	107	13
5	$\mathbf{47\,176\,870}$	$\mathbf{4098}$
6	$> 10 \uparrow\uparrow 15$	?

Tight results are available only for $n \leq 5$.

Beyond that, we only have ridiculously large bounds.

The Obstructions

There are several fundamental difficulties in computing busy beaver numbers, even for annoyingly small numbers of states.

- Brute-force search quickly becomes infeasible (try $k = 10$).
- The Halting Conundrum:
  Even if we could somehow deal with combinatorial explosion, we don't know if a machine will ever halt—it might just run forever.
- Reasoning about the behavior of Turing machines in a formal system like Dedekind-Peano arithmetic or even Zermelo-Fraenkel set theory is necessarily of limited use.

BB-5 Project

The 5-state champion was discovered in 1989 by Marxen-Buntrock but it took till 2024 to prove optimality.

The final proof was a team effort and required a lot of work. Arguably, without the **Math Toolchain** this project would be inconceivable.

See Quanta for an informal description of the project, and bbchallenge for all the details, including a proof formalized in Coq. Xu handles a particularly difficult case.

BB-5 seems rock-solid, but, in general, busy beaver results may not be quite as robust as one would like them to be, see Harland 2016.

But How?

First, use symmetries to cut down on the number of machines that need to be considered.

Run these machines in a clever way to weed out all those that clearly are not candidates: they halt too soon, or they obviously diverge. This part can be fully automated.

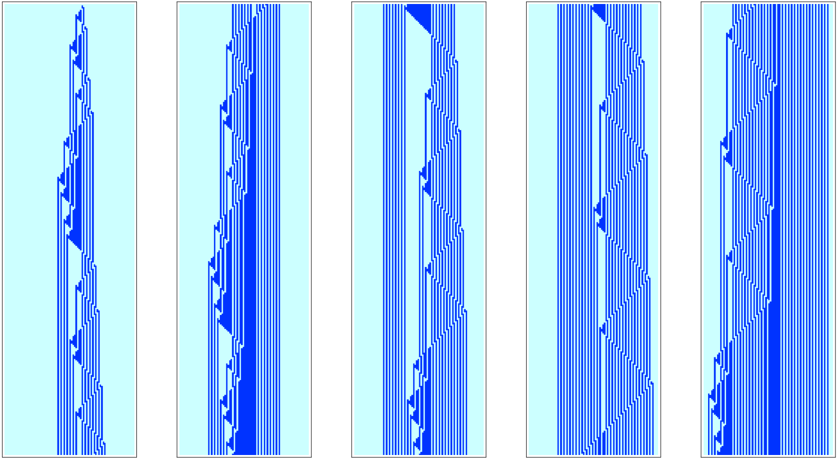
For the remaining ones, painstakingly establish divergence or run them to completion. These machines require direct human intervention.

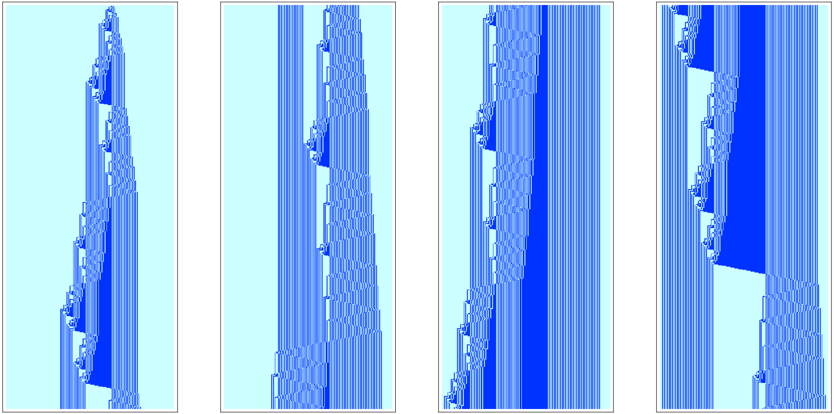
Visualization to the Rescue

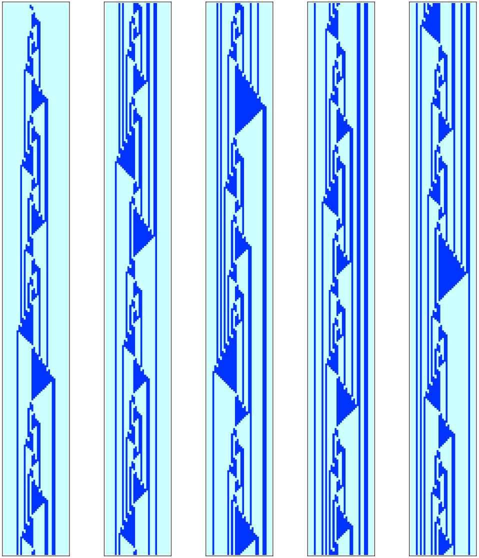
If a machine passes all the simple tests, the next step is to visually inspect the orbit.

Dire Warning: Recall the Marxen-Buntrock machine. The first, say, 1,000,000 steps of the orbit definitely look like a divergent computation—and yet it halts.

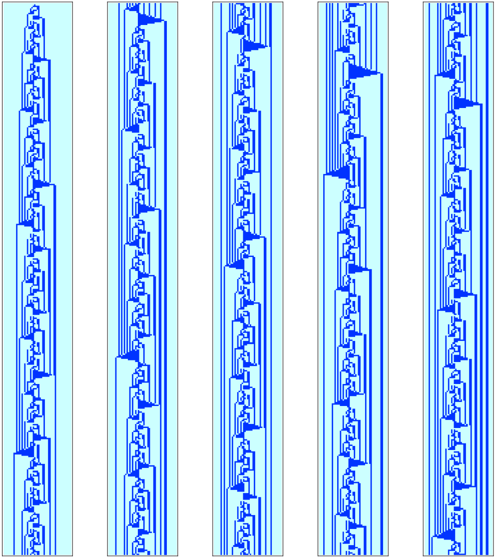
- To establish divergence we need a strict proof.
- To run to completion it helps to be able to speed up the computation.



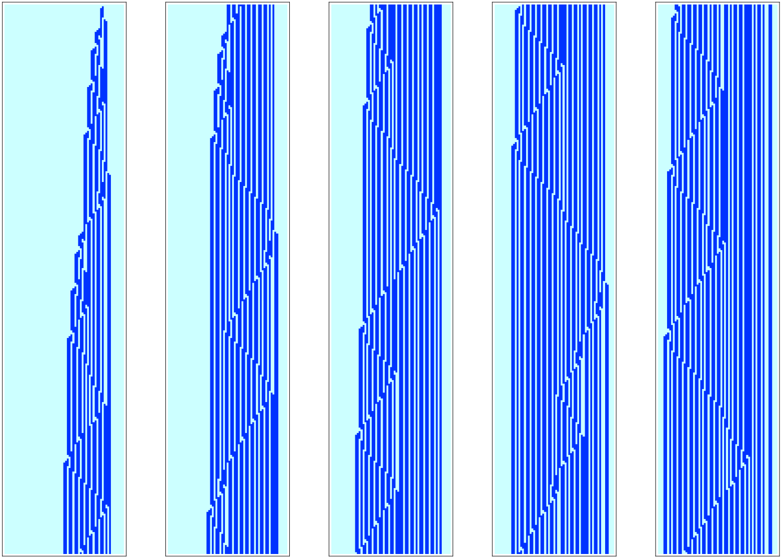




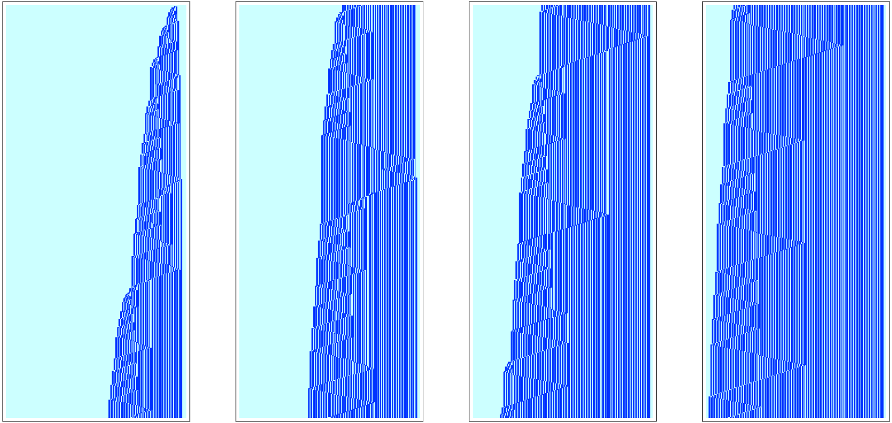
m14263231 Compressed



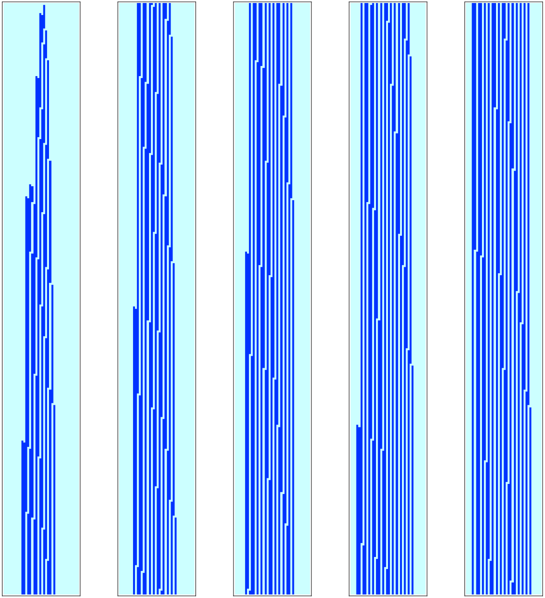
Skelet 1



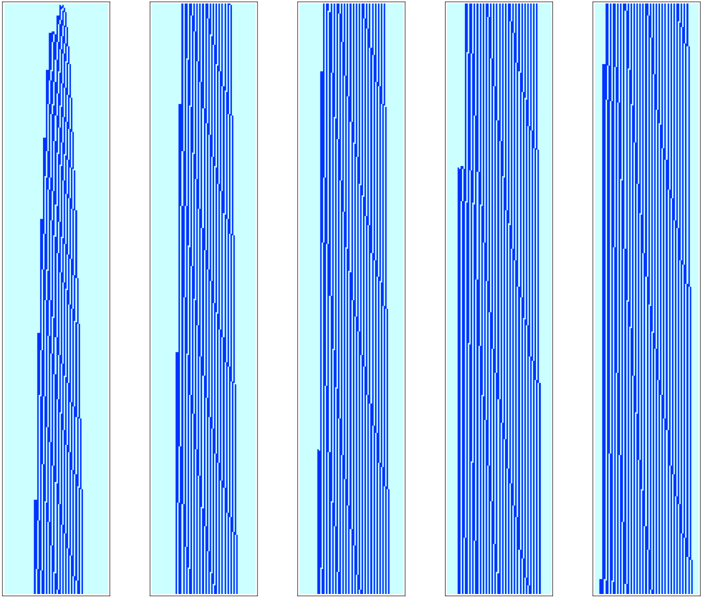
Skelet 1 Compressed



Skelet 17



Skelet 17 Compressed



For $n = 6$ all hell breaks loose.

The raw search space here has size $59\,604\,644\,775\,390\,625$, though this can be improved a bit exploiting symmetries and reachability.

Halting gets very messy here: the orbits get much more complicated and there is no good heuristic to come to the conclusion that a machine will never halt and can thus be dismissed from the competition.

A new machine by Pavel Kropitz, takes about $10 \uparrow\uparrow 15$ steps to halt. A more precise bound is

$$10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{10^{4.023873729}}}}}}}}}}}}}}}$$

You're welcome.

1 Hunting Busy Beavers

2 Towards a Proof

3 Goodstein Sequences

Ignoring symmetries, there are $(4n+1)^{2n}$ n-state machines (actually, n plus one halting state which carries no outgoing transitions).

The transition function has the form

$$\tau : [n] \times \mathbf{2} \longrightarrow [0, n] \times \mathbf{2} \times \mathbf{2}$$

For $n = 5$ there are $16,679,880,978,201$ machines, so the first order of business is to exploit isomorphs to cut down on that number.

Easy Stuff

We may safely assume the following:

- 1 is the initial state
- 0 is the halting state
- new states are encountered in strictly increasing order

By symmetry, we may assume that the first transition is

$$1, 0 \mapsto 2, 1, R$$

In the search one has to deal with partially specified machines. By a k-machine we mean a partial function

$$\tau : [k] \times \mathbf{2} \nrightarrow [k] \times \mathbf{2} \times \mathbf{2}$$

In particular we start with the 2-machine

$$\tau_0 = \{1, 0 \mapsto 2, 1, R\}$$

We are only interested in computations on the empty tape $\mathbf{0}$, so τ_0 will take one step and then get stuck.

Given a total TM τ, define its core to be the partial sub-machine that is actually used during the run on $\mathbf{0}$.

Observation: The core of τ has at most one halting transition.

Note that the n-state BB has to be an n-machine: if it were a k-machine for $k < n$ we could add more steps.

This suggests to start at τ_0 and systematically add transisions to generate all relevant machines.

Define a frame as a triple (τ, t, C) where $C^{\text{init}} \left|\frac{\tau}{t}\right. C$.

Sparse Enumeration

Start with the initial frame $(\tau_0, 0, \mathbf{0})$.

Given a frame (τ, t, C) do the following:

- If τ is defined on C, update the frame to $(\tau, t{+}1, \tau(C))$.
- Otherwise let $\tau(p, a)\uparrow$ and extend τ to τ' by:
  - A halting transition, record τ' and $t{+}1$ as halting.
  - If τ is a k-machine, use all possible transitions $p, a \mapsto \ell, b, D$ where $\ell \leq \min(k+1, n)$ for τ'.

    Explore all the frames (τ', t, C).

Comments

This is **very sketchy**.

It is critical to add some mechanism to handle divergence, as written the enumeration algorithm nevers stops.

We could stop if a machine reaches the same configuration twice, and declare it non-halting. Alas, checking for such loops requires memory, we have to keep track of the history of a computation.

Also, if we have constructed a full table without any halting instructions, the machine is non-halting.

The Last Transition

Suppose we have a frame (τ, t, C) with $2n-1$ transitions.

If $\tau(C)\uparrow$, we can

- Add a halting transition and and get a potential BB.
- Add a non-halting transition and declare the machine to be non-halting.

So the real interesting cases will be the ones where a machine with $2n-1$ transitions keeps running for a long time.

At some point we have to decide whether to either try to prove that the machine halts (and compute their halting time), or we could attempt to find a proof that it diverges.

If we just wish to verify a Busy Beaver champion (rather than to discover it in the first place), we can run plausible machines for the current maximum number of steps. The ones the have not yet been declared non-halting need to be proven to diverge.

If we have a $n-1$-machine that runs for more than $\mathsf{BB}_{\mathrm{H}}(n-1)$ steps we can declare it non-halting.

There is a basic result in computability that a Turing machine can always be made faster by a constant.

The argument goes like this: suppose τ uses alphabet Σ.

Define a new machine τ' on alphabet $\Gamma = \sigma^k$ for some suitably large k.

The new machine then can simulate at least k steps of the old one in just a single step.

Note that this is entirely useless in practice, we cannot use an alphabet $\mathbf{2}^{1000}$. Also, it is quite tricky to really determine the transitions in τ'.

Scaling Down

We can get some mileage out of a different approach, though: write the configurations in run-length encoding:

$$1^{e_1}0^{e_2}1^{e_3}\dots1^{e_{r-2}}0^{e_{r-1}}1^{e_r}$$

It may be the case that, if the machine enters a block 1^e in state p it will later leave the block at the other end in state q.

Or it might change the block to 1^{e+1}.

A smart simulator could handle this in (essentially) one step. Strictly speaking we need $\log e$ steps, but for at least for $n \leq 5$ all relevant values of e are small.

1 Hunting Busy Beavers

2 Towards a Proof

3 **Goodstein Sequences**

A special form of primitive recursion is iteration: applying a function over and over again.

Suppose $f : A \to A$ is some endofunction. Define

$$F(0, x) = x$$
$$F(n^+, x) = f\big(F(n, x)\big)$$

$F(n, x)$ is usually written $f^n(x)$.

Iteration is essentially the bottom-up version of primitive recursion and has the same computational power.

Here is an example where iteration of a (slightly bizarre) arithmetic operation produces a totally perplexing result.

Suppose we write a number in base 2, say

$$266 = 2^8 + 2^3 + 2$$

We can turn this into the hereditary binary expansion by writing the exponents also in base 2, and so on.

$$266 = 2^{2^{2+2^0}} + 2^{2+2^0} + 2^0 + 2^0$$

In the interest of sanity, we'll write 1 for 2^0, and 2 for $1+1$.

Note that there are no multiplicative coefficients, we only use addition, exponentiation and digit 0.

Base Bump

Now suppose we replace 2 in the representation everywhere by 3:

$$3^{3^{3+1}} + 3^{3+1} + 3$$

Evaluating we get a much larger number:

$$443426488243037769948249630619149892887 \approx 4 \times 10^{38}$$

Next, we bump the base to 4 and get something like 3×10^{616}.

Clearly, this process leads to a very rapidly increasing sequence of numbers.

Less Informally

Let $b \leq 2$ and $n \in \mathbb{N}$. We write

$$E(b, n) = \text{ hereditary base } b \text{ expansion of } n$$

The expression is constructed from the base b, using only addition, exponentiation, and digit 0.

For example

$$E(3, 7625597485042) = 3^{3^3} + 3^3 + 3^3 + 3^0$$

Numerical Base Bump

We write $\mathsf{bb}(b, n)$ for the numerical version of base bump:

$$\mathsf{bb}(b, n) = \mathsf{eval}\big(E(b, n)[b \mapsto b{+}1]\big)$$

For example, $\mathsf{bb}(3, 7625597485042)$ comes out to be

$$\begin{array}{l}
13407807929942597099574024998205846127479365820592393 \\
37772356144372176403007354697680187429816690342769 0 \\
0318581864860508537538828119465699464336490060840 9
\end{array}$$

About 1.34078×10^{154}.

Given a natural number n, define its Goodstein sequence $g(n,k)$ for $k \geq 1$ as follows.

$$g(n,1) = n$$
$$g(n,k) = \begin{cases} \mathsf{bb}(k, g(n,k-1)) - 1 & \text{if } g(n,k-1) \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

One can check that g is a primitive recursive function.
It is a character building exercise to do this in some detail.

Sadly, it is very hard to come up with good examples.

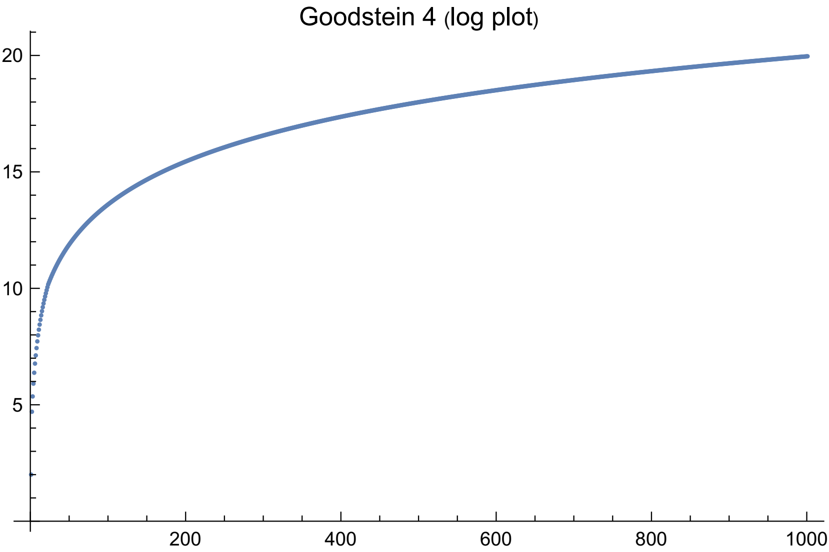
Starting at 2 and 3 we get the short sequences

$$\begin{array}{ll} n = 2 & 2, 2, 1, 0 \\ n = 3 & 3, 3, 3, 2, 1, 0 \end{array}$$

But starting at $n = 4$, things already spin out of control:

$$4, 26, 41, 60, 83, 109, 139, 173, 211, 253, 299, \ldots, 402653183^2, \ldots$$

Huge surprise: after $10^{121,210,695}$ steps we get to 0.



Goodstein 4 (log plot)
20
15
10
5
200
400
600
800
1000

The base bump operation is a strange mix of manipulating formal terms and actual arithmetic:

To replace base b by $b+1$ we need to first come up with an expression $E(b,n)$ that is based on

$$\text{addition, exponentiation, } 0$$

such that $\mathsf{eval}\big(E(b,n)\big) = n$.

Then the next element is by a simple predecessor calculation:

$$\mathsf{eval}\big(E(b,n)[b \mapsto b{+}1]\big) - 1$$

Insane Conjecture

Conjecture: All Goodstein sequences end in 0.

Of course, your intuition tells you that this is total nonsense.

To wit, the base bump increases the number by a huge amount, and then we just subtract 1, so the final result will be a tiny little bit smaller than with a pure base bump.

Obviously, the base bumps win, the sequence will diverge.

But Why?

How could this possibly work? Because once the base is sufficiently large, we keep chipping away at the constant term until we ultimately have to borrow from one of the previous terms.

Consider the sequence for $n = 4$. When we get to base $b = 402653183$, the sequence locally looks like

$$\begin{aligned} n_{b-1} &= b^2 \\ n_b &= b(b+1) + b \\ n_{b+1} &= b(b+2) + b - 1 \\ n_{b+2} &= b(b+3) + b - 2 \\ &\dots \\ n_{2b} &= b(2b+1) \end{aligned}$$

The red part is the base at that point. Ponder deeply.

Here is a rather important ordinal:

$$\varepsilon_0 = \omega^{\omega^{\omega^{\cdot^{\cdot^{\cdot}}}}}$$

This is the first fixed point of the map $\alpha \mapsto \omega^\alpha$.

Note that ε_0 is closed with respect to the operations addition, multiplication and exponentiation.

It is somewhat difficult to get an intuitive sense of ε_0, but Cantor proved a nice theorem about it.

Cantor Normal Form

For ordinals $\alpha < \varepsilon_0$, Cantor has established the following normal form:

$$\alpha = \omega^{\alpha_1} + \omega^{\alpha_2} + \ldots + \omega^{\alpha_k}$$

where $\alpha > \alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_k \geq 0$.

Given this constraint on exponents, the normal form is indeed unique.

Without the condition $\alpha > \alpha_1$ the normal form holds in general, but is typically less useful.

For example, $\varepsilon_0 = \omega^{\varepsilon_0}$.

Theorem (Goodstein 1944)

All Goodstein sequences converge to 0.

Sketch of proof.

For any number n, we will associate each element $g(n, k)$ in the Goodstein sequence for n with an ordinal, its shadow (we'll dump $n_1 = n$).

$$\Omega(n, k) = \mathsf{eval}\big(E(k, g(n, k{-}1))[k \mapsto \omega]\big)$$

The evaluation here proceeds over the ordinals, otherwise it's exactly the same as for natural numbers.

In fact, we are just dealing with the Cantor normal form for some ordinal $\alpha < \varepsilon_0$.

Claim: The ordinal shadows of a Goodstein sequence are strictly decreasing.

If $\Omega(n,k) = \alpha + 1$, then $\Omega(n,k{+}1) = \alpha$: the world of shadows does not distinguish between different bases.

Otherwise we have $\Omega(n,k) = \alpha + \omega^\beta$ where $\beta > 0$. We cannot subtract 1 from a limit ordinal, but there is no problem: the hereditary base $k{+}1$ expansion of n also loses the $(k{+}1)^K$ term after subtracting 1.

As an example, consider $E(k,n) = k^2$. Then bb$(k,n) = (k{+}1)^2 - 1 = (k{+}1) \cdot k + k - 1$ and the corresponding ordinals in Cantor normal form decrease from ω^2 to

$$\underbrace{\omega + \omega + \ldots + \omega}_{k} + \underbrace{1 + 1 + \ldots + 1}_{k-1}$$

□

Stopping Time

Define the stopping time $G : \mathbb{N} \to \mathbb{N}$ as the number of steps it takes to reach 0 in a Goodstein sequence:

$$G(n) = \min\big(\, k \geq 1 \mid g(n,k) = 0 \,\big)$$

Note that G is computable; in fact, the algorithm is not particularly complicated, the only messy part is to implement the transformation to hereditary expansions. Goodstein's theorem says that G is a total function.

Claim: $G(n)$ grows much, much faster than $A(n,n)$.

The proof requires induction up to

$$\varepsilon_0 = \omega^{\omega^{\omega^{\cdot^{\cdot^{\cdot}}}}}$$

Alas, ε_0-induction is already enough to prove that (PA) is consistent.
By Gödel's theorem, it cannot be proven in (PA).

Theorem (Kirby, Paris 1982)

Goodstein's theorem is not provable in Peano arithmetic.