EXCERPTED FROM

STEPHEN WOLFRAM

A NEW KIND OF SCIENCE

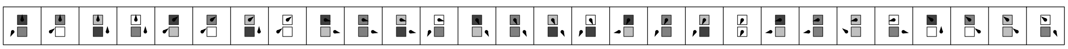
SECTION 11.12

Universality in Turing Machines and Other Systems

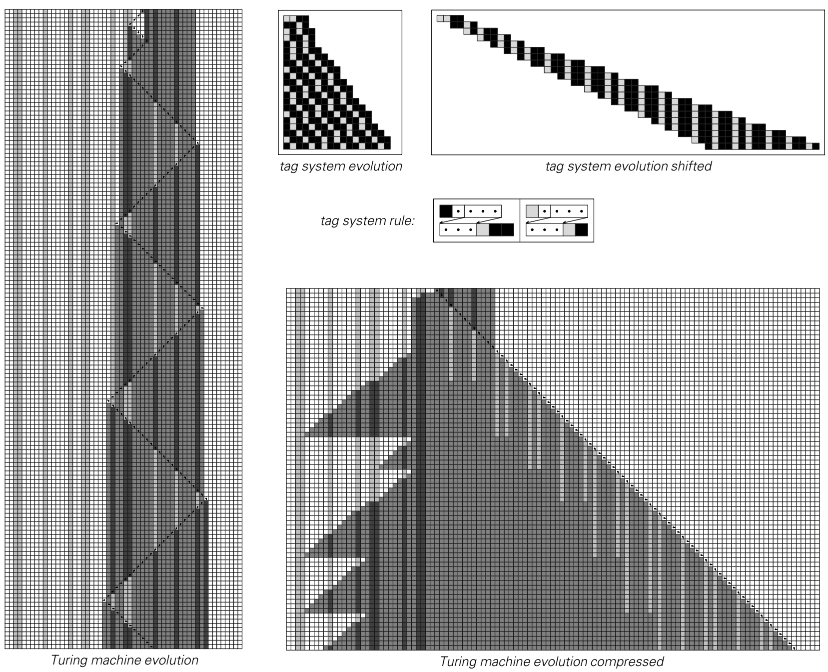
Universality in Turing Machines and Other Systems

From the results of the previous few sections, we now have some idea where the threshold for universality lies in cellular automata. But what about other kinds of systems—like Turing machines? How complicated do the rules need to be in order to get universality?

In the 1950s and early 1960s a certain amount of work was done on trying to construct small Turing machines that would be universal. The main achievement of this work was the construction of the universal machine with 7 states and 4 possible colors shown below.



The rule for a universal Turing machine with 7 states and 4 colors constructed in 1962. Until now, this was essentially the simplest known universal Turing machine. Note that one element of the rule can be considered as specifying that the Turing machine should "halt" with the head staying in the same location and same state.

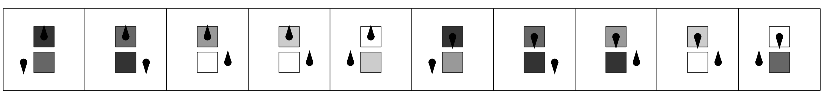


An example of how the Turing machine above can emulate a tag system. A black element in the tag system is set up to correspond to a block of four cells in the Turing machine, while a white element corresponds to a single cell.

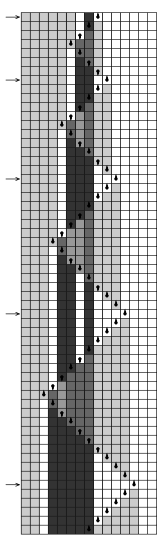
The picture at the bottom of the facing page shows how universality can be proved in this case. The basic idea is that by setting up appropriate initial conditions on the left, the Turing machine can be made to emulate any tag system of a certain kind. But it then turns out from the discussion of page 667 that there are tag systems of this kind that are universal.

It is already an achievement to find a universal Turing machine as comparatively simple as the one on the facing page. And indeed in the forty years since this example was found, no significantly simpler one has been found. So one might conclude from this that the machine on the facing page is somehow at the threshold for universality in Turing machines.

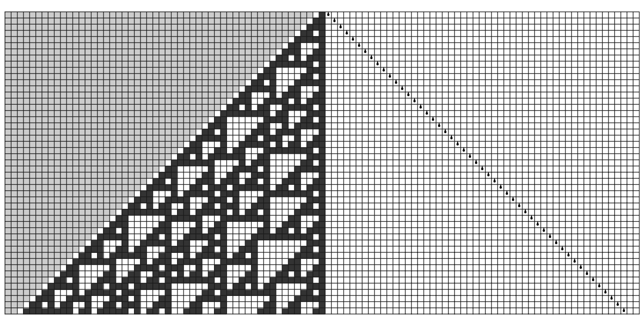
But as one might expect from the discoveries in this book, this is far from correct. And in fact, by using the universality of rule 110 it turns out to be possible to come up with the vastly simpler universal Turing machine shown below—with just 2 states and 5 possible colors.



The rule for the simplest Turing machine currently known to be universal, based on discoveries in this book. The machine has 2 states and 5 possible colors.



Turing machine evolution



Turing machine evolution compressed

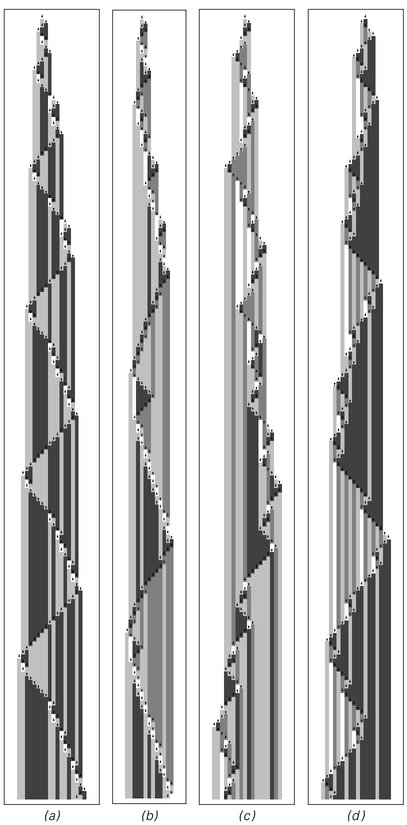
An example of how the Turing machine above manages to emulate rule 110. The compressed picture is made by keeping only the steps indicated at which the head is further to the right than ever before. To get the picture shown requires running the Turing machine for a total of 5000 steps.

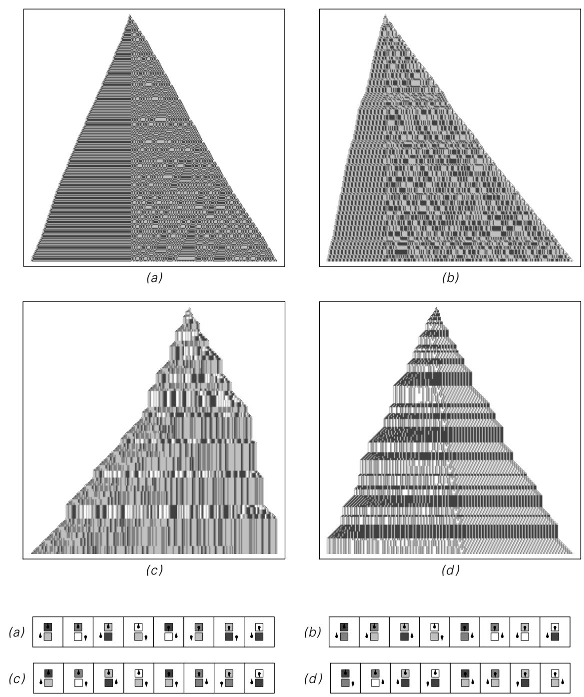
As the picture at the bottom of the previous page illustrates, this Turing machine emulates rule 110 in a quite straightforward way: its head moves systematically backwards and forwards, at each complete sweep updating all cells according to a single step of rule 110 evolution. And knowing from earlier in this chapter that rule 110 is universal, it then follows that the 2-state 5-color Turing machine must also be universal.

So is this then the simplest possible universal Turing machine?

I am quite certain that it is not. And in fact I expect that there are some significantly simpler ones. But just how simple can they actually be?

If one looks at the 4096 Turing machines with 2 states and 2 colors it is fairly easy to see that their behavior is in all cases too simple to support universality. So between 2 states and 2 colors and 2 states and 5 colors, where does the threshold for universality in Turing machines lie?





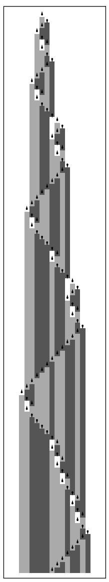
Examples of Turing machines with 2 states and 4 colors that show complex behavior. The compressed pictures above are based on 50,000 steps of evolution. In all cases, all cells are initially white.

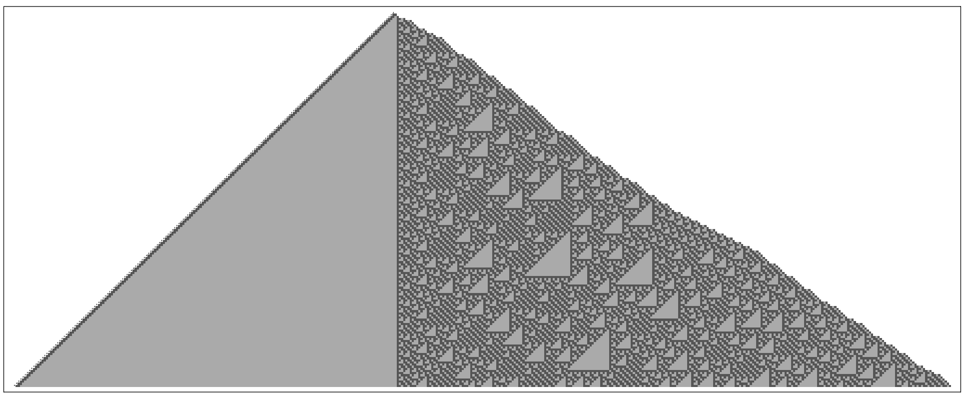
The pictures at the bottom of the facing page give examples of some 2-state 4-color Turing machines that show complex behavior. And I have little doubt that most if not all of these are universal.

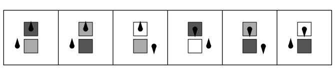
Among such 2-state 4-color Turing machines perhaps one in 50,000 shows complex behavior when started from a blank tape. Among 4-state 2-color Turing machines the same kind of complex behavior is also seen—as discussed on page 81—but now it occurs only in perhaps one out of 200,000 cases.

So what about Turing machines with 2 states and 3 colors? There are a total of 2,985,984 of these. And most of them yield fairly simple behavior. But it turns out that 14 of them—all essentially equivalent—produce considerable complexity, even when started from a blank tape.

The picture below shows an example.



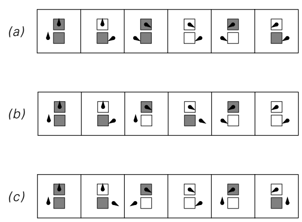




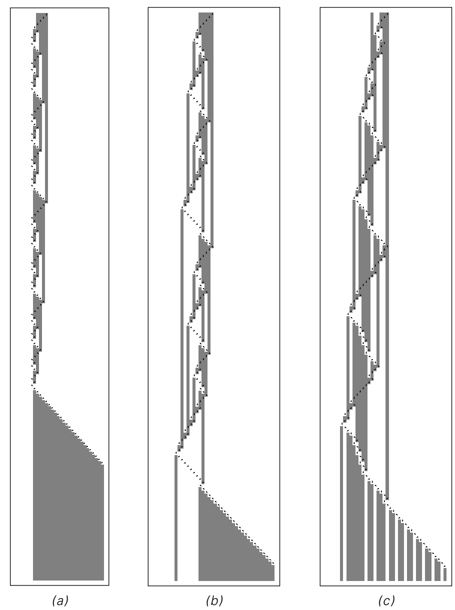
One of the 14 essentially equivalent 2-state 3-color Turing machines that yield complicated behavior when started from a blank tape. The compressed picture above is made by taking the first 100,000 steps, and keeping only those at which the head is further to the left than ever before. The interior of the pattern that emerges is like an inverted version of the rule 60 additive cellular automaton; the boundary, however, is more complicated. In the numbering scheme of page 761 this is machine 596,440 out of the total of 2,985,984 with 2 states and 3 colors.

And although it will no doubt be very difficult to prove, it seems likely that this Turing machine will in the end turn out to be universal. And if so, then presumably it will by most measures be the very simplest Turing machine that is universal.

With 3 states and 2 colors it turns out that with blank initial conditions all of the 2,985,984 possible Turing machines of this type quickly evolve to produce simple repetitive or nested behavior. With more complicated initial conditions the behavior one sees can sometimes be more complicated, at least for a while—as in the pictures below. But in the end it still always seems to resolve into a simple form.



Examples of 3-state 2-color Turing machines which behave for a while in slightly complicated ways. With more elaborate initial conditions, these machines can be made to exhibit complicated behavior for longer.



Yet despite this, it still seems conceivable that with appropriate initial conditions significantly more complex behavior might occur—and might ultimately allow universality in 3-state 2-color Turing machines.

From the universality of rule 110 we know that if one just starts enumerating cellular automata in a particular order, then after going through at most 110 rules, one will definitely see universality. And from other results earlier in this chapter it seems likely that in fact one would tend to see universality even somewhat earlier—after going through only perhaps just ten or twenty rules.

Among Turing machines, the universal 2-state 5-color rule on page 707 can be assigned the number 8,679,752,795,626. So this means that after going through perhaps nine trillion Turing machines one will definitely tend to find an example that is universal. But presumably one will actually find examples much earlier—since for example the 2-state 3-color machine on page 709 is only number 596,440.

And although these numbers are larger than for cellular automata, the fact remains that the simplest potentially universal Turing machines are still very simple in structure, suggesting that the threshold for universality in Turing machines—just like in cellular automata—is in many respects very low.

So what about other types of systems?

I suspect that in almost any case where we have seen complex behavior earlier in this book it will eventually be possible to show that there is universality. And indeed, as I will discuss at length in the next chapter, I believe that in general there is a close connection between universality and the appearance of complex behavior.

Previous examples of systems that are known to be universal have typically had rules that are far too complicated to see this with any clarity. But an almost unique instance where it could potentially have been seen even long ago are what are known as combinators.

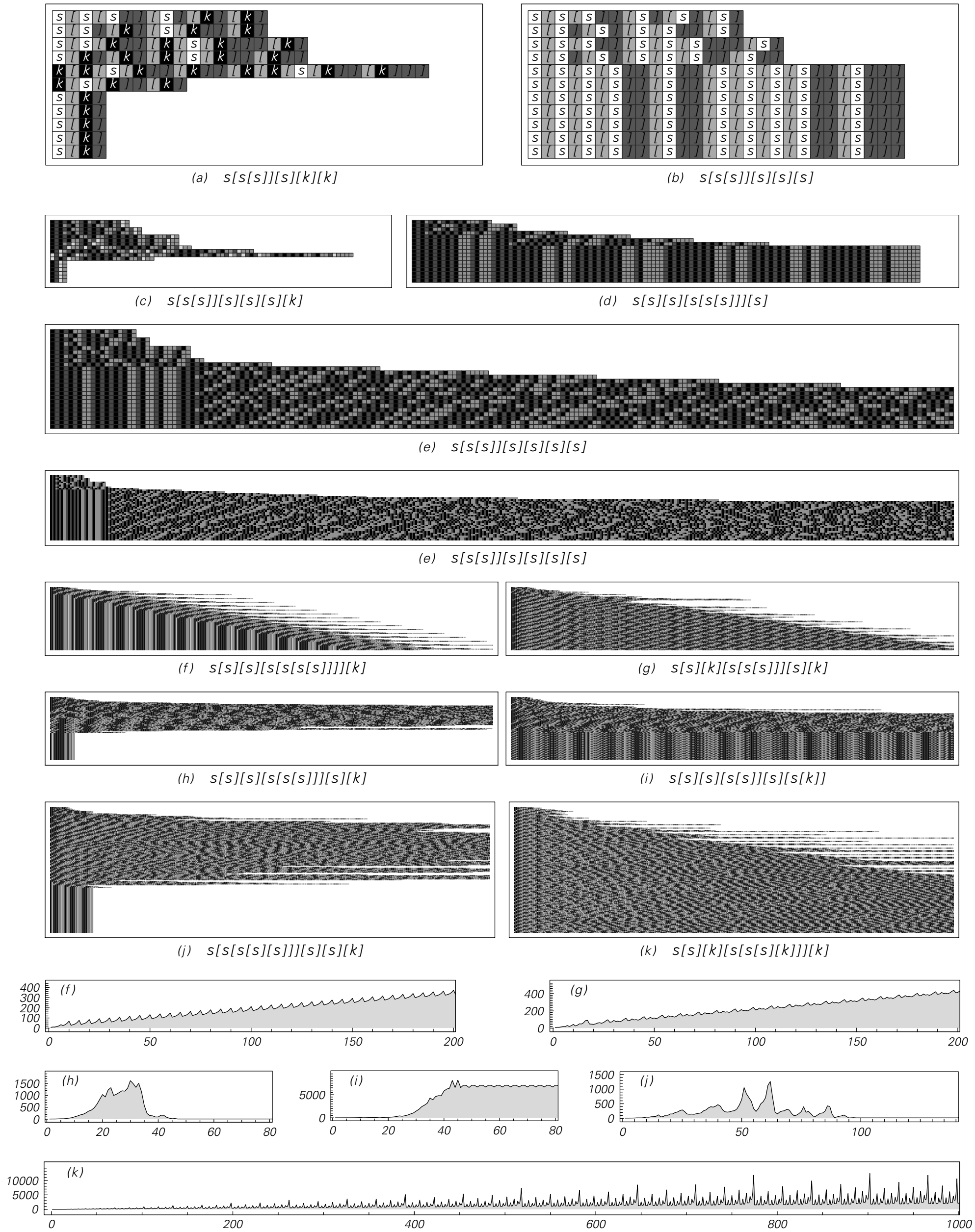
Combinators are a particular case of the symbolic systems that we discussed on page 102 of Chapter 3. Originally intended as an idealized way to represent structures of functions defined in logic, combinators were actually first introduced in 1920—sixteen years before Turing machines. But although they have been investigated somewhat over the past eighty years, they have for the most part been viewed as rather obscure and irrelevant constructs.

The basic rules for combinators are given below.

```
s[x_][y_][z_] → x[z][y[z]]
k[x_][y_] → x
```

Rules for symbolic systems known as combinators, first introduced in 1920, and proved universal by the mid-1930s.

With short initial conditions, the pictures at the top of the next page demonstrate that combinators tend to evolve quickly to simple fixed points. But with initial condition (e) of length 8 the pictures show



Examples of combinator evolution. The expression in case (e) is the shortest that leads to unlimited growth. The plots at the bottom show the total sizes of expressions reached on successive steps. Note that the detailed pattern of evolution—though not any final fixed point reached—can depend on the fact that the combinator rules are applied at each step in *Mathematica* /. order.

that no fixed point is reached, and instead there is exponential growth in total size—with apparently rather random internal behavior.

Other combinators yield still more complicated behavior—sometimes with overall repetition or nesting, but often not.

There are features of combinators that are not easy to capture directly in pictures. But from pictures like the ones on the facing page it is rather clear that despite their fairly simple underlying rules, the behavior of combinators can be highly complex.

And while issues of typical behavior have not really been studied before, it has been known that combinators are universal almost since the concept of universality was first introduced in the 1930s.

One way that we can now show this is to demonstrate that combinators can emulate rule 110. And as the pictures on the next page illustrate, it turns out that just repeatedly applying the combinator expression below reproduces successive steps in the evolution of rule 110.

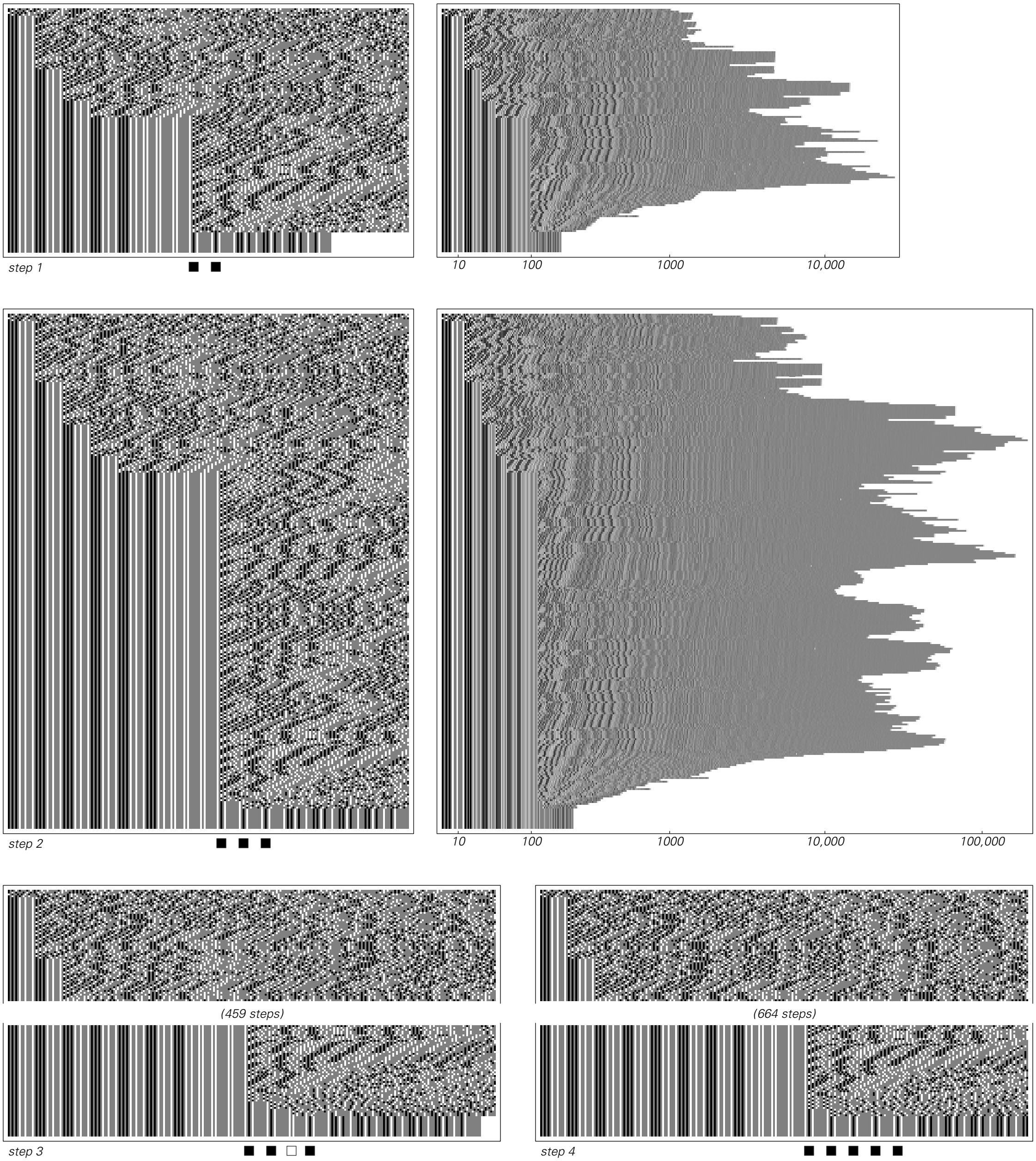
```
s[s[k[s]][s[k[s[s[k][k]]]][s[k[k]][s[s[s[s[k][k]][k[s[k]]]][k[s[s[k[s]][s[k[s[s[k][k]]]][s[k[k]][s[s[k[
s]][s[k[s[s[k][k]]]][s[k[k]][s[s[k][k]][k[k]]]]]][s[k[k]][s[s[s[k][k]][k[s[k]]]][k[s[k]]]]]]]][s[k[k]][s[s[
s[k][k]][k[s[k]]]][k[k]]]]]]][s[k[s[s[s[k][k]][k[s[s[s[k][k]][k[s[s[s[k][k]][k[k]]][k[k]]]]][k[k]]]]]]][s[k[
k]][s[s[s[s[s[k][k]][k[s[k]]]][k[s[s[k[s]][s[k[s[s[k][k]]]][s[k[k]][s[s[s[k][k]][k[k]]][k[k]]]]]][s[k[k]][
s[s[k[s]][s[k[s[s[k][k]]]][s[k[k]][s[s[k][k]][k[s[k]]]]]]][s[k[k]][s[s[s[s[k][k]][k[k]]][k[s[k]]]][s[s[s[
s[k][k]][k[k]]][k[k]]][k[s[k]]]]][s[s[s[s[s[s[k][k]][k[k]]][k[k]]][k[s[k]]]][k[s[k]]]][s[s[s[s[s[s[k][
k]][k[k]]][k[k]]][k[k]]][k[s[k]]]][k[s[k]]]][k[k]]]]]]]]]]][s[s[k[s]][s[k[s[s[k][k]]]][s[k[k]][s[s[k[s]][s[
k[s[s[k][k]]]][s[k[k]][s[s[k][k]][k[k]]]]]][k[k[k]]]]]]][k[k[k]]]]][k[s[k]]]]]]][k[k]]]]]][s[k[k]][s[k[s[s[
k[s]][k]]]][s[s[k][k]][k[s[k]]]]]]
```

A combinator expression that corresponds to the operation of doing one step of rule 110 evolution.

There has in the past been no overall context for understanding universality in combinators. But now what we have seen suggests that such universality is in a sense just associated with general complex behavior.

Yet we saw in Chapter 3 that there are symbolic systems with rules even simpler than combinators that still show complex behavior. And so now I suspect that these too are universal.

And in fact wherever one looks, the threshold for universality seems to be much lower than one would ever have imagined. And this is one of the important basic observations that led me to formulate the Principle of Computational Equivalence that I discuss in the next chapter.



Emulating the rule 110 cellular automaton using combinators. The rule 110 combinator from the previous page is applied once for each step of rule 110 evolution. The initial state is taken to consist of a single black cell.