Rise Of The Machine

A basic knowledge of mathematical symbols will be assumed. Like sets and the $\in$ element relation or from logic, the $\rightarrow$ implication , $\wedge$ = and , $\vee$ = or , $\neg$ = not.
Absolutely no assumption of proved facts will be though and I decided to use a little formalism to avoid the pretense that a mind who never ventured into mathematics will understand what follows.
Yet, I do believe that anybody can follow all that I'll explain if enough patience is present.
This of course is a rare thing today. I encourage you to venture into the net and see what is there.
The Wikipedia math articles are an obscenity. To see this for yourself again requires a certain searching and patience to face stupidity.

Texts, sets, classes

An **A** alphabet is a fixed finite set of symbols in a given order.
In our notation, letters in **A** are lower case Latin, but digits, signs and space can be included.
So **A** = (a , b , . . , y , 0 , 1 , . . , 9 , ! , . . , ? , z) and thus "a to z" means all symbols.
Texts are any finite long combinations from the alphabet. Ex: apple? no!
Texts will be referred to by capital letters A , B , . .
Sets are short for sets of texts and these will be denoted by the cursive capitals $\mathcal{S}$, $\mathcal{U}$, $\mathcal{V}$.
Classes will mean any sets of text sets. These will be denoted by the cursive capitals $\mathcal{C}$, $\mathcal{E}$.

$\mathbf{A}^{\infty}$ is the set of all texts made from **A**.

Surprisingly, $\mathbf{A}^{\infty}$ is merely a sequence of texts.

To see this, we can even use the already given sequencing of **A** as alphabetical order.
Dictionaries list words alphabetically too but this wouldn't work for all texts because we would never get to letter b since there are already infinite many texts starting with a.
The heuristic idea is to mix the alphabetical order with length order!
So we start with the 1 length texts, that is single symbols. These are simply our alphabet.
Then we list all 2 length texts alphabetically, then the 3 length ones and so on.
From the **A** = (0 , 1) simplest binary alphabet the list of all texts is this:
0 , 1 , 00 , 01 , 10 , 11 , 000 , 001 , 010 , 011 , 100 , 101 , 110 , 111 , 0000 , . . .
A nice visualization behind the sequencing of all texts is the "text tree".
We write the increasing length groups as lines under each other. So our first line is the alphabet.
Under every symbol we can start branchings down into every symbol again.
So we'll have m , m^2 , m^3 , . . increasing long lines.
The missing tip of the tree is the Δ empty text and we can draw branchings from Δ to the symbols. And indeed, every symbol is a continuation of this empty text.
Later we'll use this abstraction for two more substantial meanings behind.

A similar surprise of sequencability is to sequence all the fractions between 0 and 1.
To be surprised, you should know that these fractions on the interval [0,1] are a dense subset.
Between any two fractions lies a third because the middle of them is a fraction again.
Indeed, it can be calculated as their average.
Thus of course there are infinite many fractions between any two. So now here is the surprise:

$$\frac{1}{2} \ , \ \frac{1}{3} \ , \ \frac{2}{3} \ , \ \frac{1}{4} \ , \ \frac{2}{4} \ , \ \frac{3}{4} \ , \ \frac{1}{5} \ , \ . \ . \ . \ .$$

Even more surprising is how easy the sequencing was. We went by increasing denominators.
Cantor realized that all the points of [0,1] is not sequencable. So, for any sequence of points there, at least one point of [0,1] must be missing from the sequence.

This can be proved by realizing some basic features of how points are in the continuity.
But a simplest proof is by identifying the points with the infinite decimals.
This identification is not perfect because: .270939999999 . . . = .27094000000 . . .
So we have two decimals for some points and thus proving that the decimals are not sequencable could just mean that these double forms caused this. This however is impossible because these double marked points are only a sequence. Indeed, .1999. . , .2999. . , .3999. . , . . . with all increasing numbers before the 9-s will give all extra decimals.
So if all points were sequencable as P_1 , P_2 , . . then adding these extra all 9 ending forms would be sequencable too as P_1 , .1999. . , P_2 , .2999. . , P_3 , .3999. . , . . .
So we only have to show that for any list of decimals one is missing. Let one be:

D_1 = .2680. . .
D_2 = .7048. . .
D_3 = .6593. . .
.
.

The diagonal D decimal is formed by picking the n-th digit from D_n.
In our case it starts as D = .209. . .
It would be very strange if this D were in our list but nothing forbids this.
Now let's alter every digit of D, or to be specific let's add 1 to all digits but for 9 meaning as changing it to 0. If this decimal is D^+ then in our example it starts as D^+ = .310. . .
This D^+ must be missing from our list.
Indeed, it can not be D_1 due to the differing first digit. It can not be D_2 due to the differing second digit, and so on.

2^{A^∞} is the class of all text sets that can be made from **A**.
This total class of all text sets is not sequencable.
We already showed that all possible texts are sequencable as:
A_1 , A_2 , A_3 , . . .
So any infinite binary sequence 0010001101. . . could be used to define a subset by picking the T_n text if the n-th digit in this sequence is 1.
Thus, it is enough to show that these binary sequences are not sequencable.
For them the Cantor diagonal method again shows that D^+, which is there the opposite of D, that is changing all 0-s to 1 and 1-s to 0-s, is missing from any sequence.

Basic Picture

Cantor's set concept and the existence of different infinites caused a revolution of mathematics.
Sets as collections without any given order may actually seem as an idiotic idea. Why would we ignore the structure of a collection? Well, this can only be explained by the end result.
Indeed, at the end these very hollow collections are able to describe all structures.
The prelude to the different infinites is that many structurally different sets are actually same infinites like the naturals and the fractions. The even prior prelude to these surprises is the simple fact that an infinite set remains the same infinity by taking away only a finite amount from it.
Strangely this becomes only evident if we do regard an order. From the naturals for example we can take away the numbers up to 5 and then the remaining set 6 , 7 , 8 , . . is equivalent to the original 1 , 2 , 3 , . . This is so obvious that we are not even surprised by seeing it.
The fact that half of this set, like the odds or evens are also the same infinity is again trivial by:
1 , 3 , 5 , . . or the remaining 2 , 4 , 6 , . . are both equivalent with 1 , 2 , 3 , . .

Galileo was surprised by this when he formulated his falling law as the odd consecutive falling distances. The total falling distances from the top are thus 1 , 1+3=4 , 1+3+5=9 , . . the squares.
This is a much better law because then we can use any t time and calculate the fallen distance.
Just like I was wandering away from the infinite sets by these, he was wandering away from the fall by his observations about the infinity of the odds.
In truth, when we wander away and get surprised by something then we actually visit the future!
Imperfect glimpses of future importances are revealed to us. And all our recognitions are merely glimpses! Even such robust recognition like the set concept by Cantor hid consequences that he could not foresee. And such is the very subject of this article.
The different infinites only come in as marginal importance for us.
We made the consequence that the set of all text sets is not sequencable.
The title is about machines and so as expectable, we are going to use machine collectable sets.
Since a particular machine is a finite variation of its parts or the framework we use, thus the machines could be sequenced just like the texts. So machine collectable sets are only a sequence.
Thus, we at once know at this very start that some text sets are not machine collectable.
But this is not important for us! We have a much more surprising direction. Namely, that some machine collectable sets are such that the texts outside the set, the complement of the set, is not collectable by any machine at all. And this has an abstract side of it even without going into machines just merely accepting the set concept but used for texts.
This is a layer of the magic onion that Cantor could not foresee though it is so simple.
Text sets of course raise the same question about the ignored orders, as sets in general do.
And here too the end result is the only convincing judgment. But strange possibilities already show that this text collection idea is a very promising direction.
To raise again the doubting question: Why would you throw texts into hats that contain them without any order? And then I say: I will not only have such stupid hats but I will have a magic hat that contains all hats combined. Then you might say: Well, you already showed that all texts are just a sequence so this is a hat too. And then I say: No, you misunderstood me. I will have a magic hat from which I can recover every single hat. So they are not melted in, like in the total set of texts. And I hope this raises your attention.
The idea is so simple! Suppose you are an old fashioned mathematician like Kronecker was, who wanted to silence Cantor and also said that only the natural numbers were created by God.
So we want to collect natural numbers like say the squares or the primes. Watch this:
squares:1 , squares:4 , squares:9 , . . primes:2 , primes:3 , primes:5 , . .
Now throw all these into my magic hat. You want to get the squares? All you have to do is search in my magic hat for the beginning “squares:” and cut this label off. You will get the squares. But you will also get the primes and any other number collections. Now if you are a notorious doubter with intelligence then you will come up with this: Well, I bet you will not be able to find your magic hat collection in your magic hat too. And so here is the punch line: Sure I will!
Imagine that we have labeled and threw into my magic hat everything except my magic hat itself.
Now keep all that is there but also add as beginning to every text “magic hat:” and add these texts too. I know, this is not good because cutting off “magic hat:” will be only the full previous set but not the magic hat itself. So I am not finished. I will add not only “magic hat:” but the double, triple and all repeats of this label as prefix to all texts. Then indeed, taking off “magic hat:” from all texts that contain it, will give back the full magic hat set.
So we gave a twist to the fact that infinite sets, in particular already an finite sequencable set, can almost contain itself. The “almost” at the equivalences meant merely a one to one possibility.
But here we truly feel that the magic hat is inside itself which of course is impossible.
We have to regard a proper subset, namely the texts that start as “magic hat:”. But then cutting this label off, indeed will get back the original full set.

Basic Theorems

$\neg\mathcal{S}$ = { T ; T $\notin$ $\mathcal{S}$ } = the complement of $\mathcal{S}$.
BE = the B text continued with E. ex: applecider made from B = apple and E = cider.
B + $\mathcal{S}$ = { BE ; E $\in$ $\mathcal{S}$ }.
$\mathcal{S}$ – B = { E ; BE $\in$ $\mathcal{S}$ }.
$\mathcal{S}$/ 2 = { T ; TT $\in$ $\mathcal{S}$ }.
$\mathcal{U}$ is a universal set in the $\mathcal{C}$ class if $\mathcal{U} \in \mathcal{C}$ and for any $\mathcal{S} \in \mathcal{C}$, there is a B text that:
$\mathcal{U}$ – B = $\mathcal{S}$.
It may feel paradoxical that using as $\mathcal{S}$ the $\mathcal{U}$ universal set itself, we get $\mathcal{U}$ – U = $\mathcal{U}$ for some U text. But this is very possible. In fact, if any $\mathcal{U}$ set contains texts that don't have a U as beginning, then adding to $\mathcal{U}$ all its texts again but with added U , UU , . . beginnings, we will get a $\mathcal{U}^*$ where the U continuing texts are exactly the same set as the full $\mathcal{U}^*$.
An $\mathcal{S}$ set shares the T text with the $\mathcal{V}$ set if either T is element of both or neither of them.
So in other words if T $\in$ $\mathcal{S}$ $\leftrightarrow$ T $\in$ $\mathcal{V}$.
Obviously, the only set that shares no text at all with an $\mathcal{S}$ set is $\neg\mathcal{S}$.
Now here are the basic four theorems:

1. If $\mathcal{S} \in \mathcal{C}$ and $\mathcal{S}$ shares text with every $\mathcal{V} \in \mathcal{C}$ then $\neg\mathcal{S} \notin \mathcal{C}$.
2. If $\mathcal{U}$ is universal set in $\mathcal{C}$ and $\mathcal{U}/2 \in \mathcal{C}$ then $\neg$ ($\mathcal{U}/2$) $\notin \mathcal{C}$.
3. The following three can not stand together:
   a. $\exists$ $\mathcal{U}$ universal set in $\mathcal{C}$
   b. $\mathcal{V} \in \mathcal{C} \rightarrow \mathcal{V}/2 \in \mathcal{C}$
   c. $\mathcal{V} \in \mathcal{C} \rightarrow \neg\mathcal{V} \in \mathcal{C}$
4. There is no universal set in the 2^{A^∞} class of all text sets.

1. is trivial because $\mathcal{S}$ shares no text with $\neg\mathcal{S}$.
2. By 1. Enough to show that $\mathcal{U}/2$ shares text with every $\mathcal{S}$. And indeed:
   For every $\mathcal{S} \in \mathcal{C}$ there is a B text that $\mathcal{U}$ – B = $\mathcal{S}$ and thus:
   B $\in$ $\mathcal{S}$ $\leftrightarrow$ BB $\in$ $\mathcal{U}$ $\leftrightarrow$ B $\in$ $\mathcal{U}/2$.
   So $\mathcal{U}/2$ shares B with $\mathcal{S}$.
3. a. and b. are the conditions of 2. and the claim of 2. is defying c.
4. b. and c. are true for 2^{A^∞} so a. must be false.

Set assignments

A set assignment system is a [ ] function that assigns a set to any A text. So [A] = $\mathcal{S}$.
The A text can be regarded as merely a name of [A] but it may even be a rule how to produce this collection of texts. In either case we can have more A that name or produce the same set.

By our earlier remark about why the universal set is not so absurd, we might try to make a universal set for any class of sets that are assigned by a [ ] system.

Regard $\mathbf{A}^{\infty} = A_1, A_2, A_3, \ldots$ and also regard a list $U, B_1, B_2, B_3, \ldots$ that these texts are non-continuing each other. Combine all $B_n + [A_n]$ sets and also add the texts obtained by adding the $U, UU, \ldots$ beginnings to every text.
The non-continuing ensures that the B_n continuations will give $[A_n]$ and the U continuations will be the same as the full set. What we forgot about though is that we may not have this wild construction among our assigned sets. In other words, it may not be any $[A_n]$.
If one aims to get all the effective set collections then this wild construction should be there but then a more natural universality comes about by itself. The wild idea was not that wild though!
A non-continuing $B_1, B_2, B_3, \ldots$ set of texts will come about, but they themselves will produce all effective sets and thus the construction of the universal set is even simpler as combining all $B_n + [B_n]$ sets. So to the A texts that are not such B-s we could assign any effective sets. As simplest choice we'll assign the empty set.
A non-arbitrary U text will come about too that will actually name the universal set as $[U]$.
We might start to think that this U is some magic text but it's even spookier because these non-continuing texts are actually rules of collection. So U is a magic formula too.
The secrets behind this magic were unveiled by Turing. He realized first with full conviction that this $\mathfrak{E}$ class of the effective text sets exists as an objective almost physical reality. He also found the method that is the most convincing to produce all effective sets. And thirdly he realized that texts as programs are the natural collectors of texts, so an assignment comes about too.
A fourth thing that he not quite explicitly accepted was that in spite of his framework of machines or computers being the best, Effectivity is a concept that can not be defined in mathematics.
He was aware of the already ongoing dilemma about this.
To have a concept that is outside mathematics and yet is not in physics is not an easy thing to swallow. We still didn't swallow it. Will physics once embrace Effectivity?
Most amazingly, a newer similar concept emerged, Randomness.
An other important point is that it is not obvious that a mathematical definition of Effectivity is impossible. We could have something that instantly contains all possible effective methods.
The frameworks instead are very narrow and only close examinations show that they can simulate each other. This must be part of the bigger picture where Effectivity is a single concept.
To find merely examples, that is particular sets in $\mathfrak{E}$ is very easy.

A strange example is the set of all texts. We saw how $\mathbf{A}^{\infty}$ can be listed by the heuristic length increasing alphabetical order and in our wild construction of a universal set we even used this but just very formally. We even regarded a $B_1, B_2, B_3, \ldots$ text list that was supposed to contain only non-continuing texts. Try to give one concretely! If you go increasingly in lengths then you only have to make sure that no earlier members are continued. But that's not always possible! If you start as $0, 1$ you are already screwed because everything continues one of these.
If you use 0, but leave 1 open for continuation then only 1 beginning texts can come.
Using 10, 11 both again is doom. So using only 10 leaves 11 open.
It almost feels that the only non-continuing sequence is $0, 10, 110, 1110, \ldots$
But this is far from the truth. We just don't have to be so eager to list short texts.
Amazingly, not only can we create an infinity of such non continuing lists but the set of these are not even sequencable. And this can be shown again by Cantor's diagonality.
If we accept that the effective methods use only finite fix steps just like an alphabet is used in texts, then just as the texts are listable, the effective methods should be too.
So the effective text sets can not exhaust all text sets, not even the non continuing ones by what I mentioned about them being non sequencable.

Naïve effectivity

The sequencability of the full set of texts showed at once that they are an effective set.
This would suggest that effectivity should be merely regarded as sequencability.
This however is a very half-baked idea.
Cantor's sequencability concept hid effectivity behind but exactly jumping over, that is ignoring effectivity was that made him able to define the crucial non sequencability of all points.
A new name that suggest not to jump over the effectivity in a sequencing is generation.
So a generable sequence is done by some effective method. But this is still a not so good idea!
It hides an interesting duality though that we see already for number sequences. Namely, whether we give the members by some rule that uses the indices that is their places in the sequence or we use some recursive method to get the members from each other. We might combine both and think we achieved something general but we are wrong. A crucial big jump is still missing to allow potentially infinite searches for deciding if we want to list a certain number. This has been finally realized by those who followed this number generation idea too but the inherently search based method was proposed by Turing. He already used the word "computer" for this but initially the Turing machine name became accepted. Then as real computers started to become household items, even the word effective was suggested to be replaced by computable.
I regard this stupid and misleading. Actually, the computer name for the personal computers is paradoxical already. I wonder if a mother ever asked her teenage daughter why she is always in front of her computer when she has nothing to compute. But this joke has a serious side, namely how we started this whole article. With texts. Turing machines are made for texts and so the practicality of household computers as text containers merged with the purely theoretical.
I wonder if the home computers as picture containers will have some deeper meaning too.
Turing machines are text alterators letter by letter and this alteration process can run forever or stop by some inner condition. This means not most importantly the altered text as result, rather that the input starting text was "recognized". So Turing machines are recognizers.
This is the most suitable for effective collections, that is as effectivization of the set concept.
In sets we are not interested in order and the machine recognizable texts are also without any order. We supply the inputs as we wish. The machine just tells if it is recognized.
But if not, it just runs forever so we have no opportunity to ask for more inputs.
This seems like a flaw in the whole idea but not really. This input supply is an external problem.
If we imagine infinite many copy of a machine and each starting from one of all the possible texts as inputs then we will get all recognizable texts recognized and we can collect them.
What's more, every single machine can be extended to a text generator because these machines have an infinite memory! So the machine itself can generate all texts increasingly and try out each as input. Of course, the same problem arises as we giving the inputs. Namely, that if an alteration runs forever then it wouldn't get to the next one to try as input. But now the solution is this:
After it creates the next input it only does one alteration step on that. Then returns to the earlier ones and does one more step on all of them. Eventually all inputs will be altered forever or up to their stops. This generator machine will have an infinity of stops and can even display the recognized inputs as the real results. The reason we regard the recognizers as basic and not these generators is that this would merely hide the fundamentally recognition idea of text alterations.
In spite of this, already the above needed far from obvious tricks to create generators reveals that generation is a fundamental sideline of recognition. An other interesting feature hidden in generation is the lengths of texts. We don't feel any absurdity in generating our texts in increasing order and yet the so emphasized feature of recognition as potentially infinite search shows that such length increasing generations must be very primitive. Indeed, observe that in such listings of texts, every time we surpass a length we can at once list the missing ones of that length too.
So the complement set is generable at once. At recognitions with potentially infinite examinations, the whole point is that the complement is not expected to be recognizable. Indeed, why would all the failing recognitions be somehow just be recognizable by a single method. This natural expectation of non recognizable complement of course implies the same for generations.
Truly general generation should have no generable complement and so must be non increasing.

The most important common feature of recognition and generation is that they are much more mechanical than the rule systems that would seem to be the most natural examples of effectivity.
Games, equation systems, geometrical constructions and derivations in an axiom system are all "free" applications of fix rules and thus seem to be much wider than some mechanical method.
But observe the crucial point that the options are only finite. And so we can go through them in any arbitrary order. The situation is confused by the fact that mathematicians or chess champions are making intuitive decisions and this is regarded as relating to the free choices in the steps.
But this is totally false. The free steps of the rule applications are never the things about which a mathematician or chess champion thinks about. In math, the exact rule choices only come in when the idea of a proof is worked out and gets published. If a mathematician would think in terms of the axioms and rules of logic then he always had only a finite dilemma. But thinking is infinite in a substantial sense that we don't understand yet. This doesn't alter the fact that a mechanization of the "free" derivations is true and thus for example all theorems are generable. Even those that yet haven't been discovered. This sounds paradoxical but regard chess again. Though the champions are not thinking in terms of possible next steps, we can generate all possible step sequences.
The chess machines do this and thus make no mistakes. But a real machine has only a given memory to think ahead for a limited number of steps. So the deep question is whether using big enough computers, that is increasing this step number, could a machine outsmart human intuition.
I think so and yet this doesn't alter my previous belief that thinking is a new infinity.
It merely proves to me that chess is a very limited field of thinking.
So using totally mechanical recognitions that can run forever was a crucial step to see behind the "free" rule systems.

Other directions

This heuristic idea of running forever and thus not collecting a text is potentially present in Number Theory too. We can say that collect those n numbers for which "this and this" is true and we can not be sure for concrete n numbers whether the $P(n)$ property that exactifies the "this and this" stands or not. But deeper examinations of P will not be so much different for P or its negative $\neg P$ and so either both are unpredictable or both will be effective.
A bit more promising is to say collect those n numbers for which there is "such and such" other N number. If the $R(n,N)$ relation that exactifies the "such and such" is complicated enough then we shouldn't be able to predict the existence of N and so we actually have to search all numbers. But such number collections where one set is possible but the complement is not, were not discovered in the mostly dually expressed property or relational world of Number Theory.
Now, with hindsight we can create lots of number collections without collectable complement.
The lure of formalizing free derivation systems so that a universal is obtained is hard to resist too. Gödel was chasing this for a long time and finally succeeded with a very simple idea borrowed from Herbrand. And yet this theoretically so beautiful system didn't bring the important big consequences as Turing's "primitive" text alteration or Kleene's "singular" first number search.
A crucial beauty behind Turing's system as opposed to Gödel's, Kleene's or the first found by Church is that they all use number functions. This in itself could be regarded first as something heuristic as objects obtained from objects. And indeed, actually even Turing's method seems to go toward this because the text alteration temptingly offers the altered text as the "result", so object ordered to object. Only a second, heart of the matter idea will get out of this temptation and regard the original altered text as the one to be collected. But even this is not the point now. If object to object would be regarded as heuristic then the other systems would be still ugly because of the first word above that they use "number" functions. And this is still not the point. As I mentioned, Kronecker regarded the natural numbers as the God given basic reality. So now we can come to why he was not only immoral with Cantor but had a false idea about God too. The two may have had a lot to do with each other. But one thing is sure. Numbers in themselves can not give new numbers. Only more numbers can produce a new number and this is evident by the basic

operations that give a new number for any two old. Combining these then we can get a new number from more old ones as $f(x_1, x_2, \ldots, x_n)$ functions.
So every combining of the basic operations has a certain n "arity" or input number.
This is the root of ugliness in using functions and it is due to numbers being individual blocks that can not melt into each other. So even if we wanted to avoid functions and thus regard $f(x_1, x_2, \ldots, x_n)$ what it truly means, an $(x_1, x_2, \ldots, x_n, y)$ collection of tuples with merely the last elements y-s being unique, we would still have this $n + 1$ arity on our back.
So now I will offend all Number Theorists and make Gauss turn in his grave but say out loud that "natural numbers are definitely not God given fundamental blocks".
They are subjective and very imperfect blocks. This is very clear from Number Theory itself!
Primes, the so mysterious chain of pearls in the assumed God given blocks show this too.
Fermat's Last Theorem is a big exclamation mark to show us how irrelevant these are.
And don't tell me that prime exponents are enough because this is just a triviality and doesn't move to an explanation.
But I don't want to elaborate on these because have a much more convincing line and a prediction.
Paul Erdös said about a problem that "mathematics is not ready for this yet".
So first my prediction: This problem has nothing to do with primes just as Fermat's Last Theorem.
But with this much simpler mystery as I said, there is a convincing line too that shows why the solution will not be in Number Theory for sure.
So here is the problem that actually starts as a parlor trick. Think of a number! If even, divide it by 2 as many times you can to get an odd. If odd then multiply by 3 and add 1 to make it even. Then again make it odd by the first method and then again even by the second and so on as long you can go. I promise you will not have to do these infinitely because you will get down to 1 and so get into the $1 \to 4 \to 1$ infinite end cycle which is at once visible.
The literature is filled with number theoretical examinations of these so called "hail stone sequences" that come about with different starting numbers but always "fall down" to 1.
Now regard this. Write the initial number in base 2 as a binary 1001110100.
The divisions by 2 to make the number odd is now a simple cutting off all 0-s from the end.
So we get 10011101 a "1 to 1" sequence of 0-s and 1-s. Remember this for later when such same texts will come in with Turing machines. But now just check what the "evening" does.
Instead of multiplying by 3 and adding 1, we can multiply by 2 add 1 and add the number again. Multiplying by 2 is of course just placing a 0 at the end, so with placing the added 1 instead, we merely have to place a 1 to the end: 100111011. Finally, the only non trivial alteration is calculating 100111011 + 10011101 or rather

$$\begin{array}{r} 100111011 \\ \underline{10011101} \\ ?????????? \end{array}$$

The last digit of the result is definitely 0. If there is more at the end it's even better because we'll cut these off and start again. The claim is that our "1 to 1" numbers will become 1.
Observe that these can only grow if the addition overflows. Indeed, we only added one 1 and will definitely cut off a 0. And overflow comes about if and only if in the extended number with 11 at the end, the first 11 is such that there is no 00 before.
This might even sound as an instant solution but follow it and you'll see differently.
I rest my case and hope you'll agree that this problem is not about numbers rather about texts.

Randomness, the second new essence beside Effectivity

Let's return to the ugliness of arity or number tuples. Avoiding functions and trying to derive only tuple sets does give a certain clarity that I will explore a bit in the next section but first I want to mention that the texts being heuristically superior to numbers did come to light in the object to object world too. Namely, Kolmogorov realized that the text alteration results can be regarded actually as chosen texts if we try all possible alterations with all possible starts and regard the one that gets that result from a shortest possible start. This shortest input text then could be regarded as a compression of our result text. Indeed, the machine can produce our longer text from that.

Then looking at lengths, we got not just compressions but compressibility of different texts.
This then means a comparable simplicity of texts, or in negative their non-compressibility means a complexity of them.
One million alternating 0-s and 1-s can be described by only 33 symbols namely as "one million alternating 0-s and 1-s". That's not a joke, it is a perfect idea and the problem is not that we used English rather that we have to allow all machines. So then even if we regard two million 0-s or 1-s randomly instead of our alternating example, a machine can have these in its memory and voila get them in a single step making them seem simpler than the alternating 0-s and 1-s.
But this "glitch" will not defect the grand basic idea for two reasons. Firstly, because we'll have a universal machine and secondly because we'll go towards using this complexity of texts for beginnings of an infinite text to decide if it is random or not. There was still a second "glitch" even in this idea, recognized by Martin Löf. So he offered a different approach to define Randomness and then the glitch was solved and turned out to give the same definition as Martin Löf's. Halleluiah praise the Lord! Well not quite yet! If these definitions are examined more closely, alternative definitions come about that are not identical and all can pass as Randomness without trivial defects. So Randomness splintered. This is not a glitch anymore. It is a sign of something that we don't know yet. So the title of this short section makes sense.

Derivable "Cases"

Now we can come to the promised little detour to see that derivable tuple sets, in spite of the arity ugliness are nicer than deriving functions. The title is again important and reflects that something more general is here than collecting number tuples.
Indeed, every theory has basic relations but also fixed basic objects too. We can call these the names, though usually they are called constants. This idiotic name refers to that most treatments allow basic functions too and when these are not really dependent on variables because there are no variables then they are fix. But we shouldn't allow basic functions only names.
Here I strangely seem to agree with Kronecker and so regard the generation of the natural numbers as 0 , $S(0)$, $S(S(0))$, . . by a successor function as not just stupid but deceptive.
It's much nicer to admit we have an infinity of names 1 , 2 , 3 , . . as natural numbers and regard the $x \lhd z$ consecutiveness as basic relation and thus $1 \lhd 2$, $2 \lhd 3$, $3 \lhd 4$, . . as axioms.
My nephew who just enrolled to the same math high school that I went more than fifty years ago when it started, told me last week that the zero is a natural number. He said it so vehemently that I saw he had no clue why they implanted this dogma in his head. There is an anecdote about Peano who started this dogma. Once at the opera when they gathered their coats after the show, his wife said "did you get all pieces?". Peano said "yes, zero, one, two, three". The wife looked at the four coats and said "and you are the mathematician".
I agree with his wife and will start the naturals from 1.
The Peano rules of course can be stated just as well:

$$x \lhd z \;\rightarrow\; x + 1 = z$$

$$x + (y + 1) = (x + y) + 1$$

$$x \cdot 1 = x$$

$$x \cdot (y + 1) = x \cdot y + x$$

$$x^{1} = x$$

$$x^{(y+1)} = x^{y} \cdot x$$

Avoiding functions and regarding $x + y = z$, $x \cdot y = z$ and $x^y = z$ as relations just as $x \lhd z$, is easy. In fact, the rules can be stated as this:

$$\left.\begin{array}{c}(x \lhd z)\ (y = 1) \\ (x + v = w)\ (v \lhd y)\ (w \lhd z)\end{array}\right] \quad \rightarrow \quad x + y = z$$

$$\left.\begin{array}{c}(x = z)\ (y = 1) \\ (x \cdot v = w)\ (v \lhd y)\ (w + x = z)\end{array}\right] \quad \rightarrow \quad x \cdot y = z$$

$$\left.\begin{array}{c}(x = z)\ (y = 1) \\ (x^v = w)\ (v \lhd y)\ (w \cdot x = z)\end{array}\right] \quad \rightarrow \quad x^y = z$$

The lines mean $\vee$, so the different ways to get the target relation after $\rightarrow$, while the members in a line are $\wedge$ so show all the conditions necessary to get the target.
The name axioms are of course also usable and most importantly, also any already derived cases of the target relation. The derivation of $4 + 3 = 7$ is this:

$4 \lhd 5 \quad \wedge \quad 1 = 1 \qquad \rightarrow \qquad 4 + 1 = 5$

$4 + 1 = 5 \quad \wedge \quad 1 \lhd 2 \quad \wedge \quad 5 \lhd 6 \qquad \rightarrow \qquad 4 + 2 = 6$

$4 + 2 = 6 \quad \wedge \quad 2 \lhd 3 \quad \wedge \quad 6 \lhd 7 \qquad \rightarrow \qquad 4 + 3 = 7$

As we all know from elementary school, multiplication is repeated addition that is:
$x \cdot y = z$ means $x + x + \ . \ . \ . \ . + x = z$ and similarly exponentiation is repeated multiplication. The problem is of course that these dots are not exact. Neither as explicit forms nor as exact rules what to calculate. That was the whole motivation for the Peano rules.
The hidden side is that these still are not explicit so don't give the target as defined by earlier relations because the target also appears in the conditions.
Though we don't allow this in explicit definitions, we do allow this in axioms.
So these Peano rules should be regarded as simply axioms from which we get case derivations that is fully concretized name "instances" as the proper expression goes.
And this is indeed fully general for any theory. We can regard for any $B\ (x, y, \ . \ . \ . \ , z)$ basic relation the derivable $B\ (N_1, N_2, \ . \ . \ , N_n)$ cases as a $[\ B\ ]$ set of name tuples which is thus the effective meaning of B by the theory.
The heuristically beautiful side of this definition is that it instantly contains the so important one sidedness of derivations. Meaning that a complement set $\neg[\ B\ ]$ doesn't have to be $[\neg B]$ at all. It doesn't even have to be $[\ B'\ ]$ for some other B' basic relation.
Now if we have reason to believe that our system is such that every effective collection of name tuples in it are some $[\ B\ ]$, then finding a concrete $[\ B_0]$ where we are sure that its complement $\neg[\ B_0]$ is not among the $[\ B\ ]$-s, then this means that $[\ B_0]$ is an effective collection without

effective complement. And this would be crucial for other axiom systems. Simply because since Gödel we know that most complex axiom systems contain undecidable statements.
Few years later it became clear that the point is not that a particular S statement is undecidable rather that the non-theorems as set is not effective. Which means that no system can derive the non-theorems at all. This at once implies the undecidability of some S with the following heuristic argument:
Suppose all S were decidable that is S or $\neg S$ derivable. Then we had a very simple derivation system of the non-theorems by deriving the theorems and then just negate them.
A hidden assumption was that the system is consistent and so doesn't derive both S and $\neg S$.
And indeed, if our system is inconsistent then as logic is, every statement is derivable and so indeed both the theorems and the identical non theorems are derivable.
An other less obvious hidden assumption was that such formal negation is effective.
A fix axiom system becoming a good framework for Effectivity by this heuristic case collection or [B] effective meaning is obviously impossible, simply because we have infinite many effective collections and only finite many basic relations in a theory.
The solution is to allow an infinity of basic relations.
Remembering that we want all possible effective collections then suggests the following:
We should allow all possible statements, containing only formal $R (x , y , . . . , z)$ relations as targets. So only the defining statements should mark their target relations.
A fundamental problem is that a contradictory statement would make everything derivable.
So we need an avoidance of contradictory defining statements.
A case in itself is never contradictory only if the negated relation contains the same case.
So if these target relations are never negated we are okay. This of course implies that we have to be careful with quantors and logical operations too. The $\wedge$ and $\vee$ operations have a simple double layer as necessary minimum, which beautifully can be used already as the situation matrix of any prenex, that is frontally quantized statement.
There we simply write the $\wedge$-s with commas as lines and the lines are the $\vee$-s.
Our above conditions were exactly like this except we omitted the commas too.
Using the single "effective implication" from such matrix to a single relation as consequence, will at once solve all our problems! It designates the target as this consequence and interprets the quantor usage too. Namely, the target variables $x , y , z , n , . .$ are all universal while the rest of the variables $u , v , w , k , . .$ are all existential. Thus a rule:

$$\left.\begin{array}{l} x , y , z , . . \\ \\ u , v , w , . . \end{array}\right] \quad \rightarrow \quad T (x , y , z , . .)$$

actually means: $\forall x \forall y \forall z . . [\exists u \exists v \exists w . . (\text{cond}) \rightarrow T (x , y , z , . .)]$
Where (cond) is the exact $\wedge$ and $\vee$ form of our conditions.
So now all our rules above fit into this bigger picture. They are now not the axioms of the theory which is Number Theory in this case, rather part of this new extension.
They may also appear as proper axioms by coincidence.
As an interesting situation let's regard the $<$ relation. This is explicitly definable by addition as: $x < z \ : \exists y \ (x + y = z)$ but it is also definable by the effective rule:

$$\left.\begin{array}{c} (x \lhd z) \\ \\ (x < w) \quad (w \lhd z) \end{array}\right] \quad \rightarrow \quad x < z$$

The exact logical form for this rule by the aboves is:

$$\forall x \ \forall z [(x \lhd z) \vee \exists w ((x < w) \wedge (w \lhd z)) \rightarrow (x < z)]$$

The derivation of $4 < 7$ is this:

$4 \lhd 5 \quad \rightarrow \quad 4 < 5$

$4 < 5$ and $5 \lhd 6 \quad \rightarrow \quad 4 < 6$

$4 < 6$ and $6 \lhd 7 \quad \rightarrow \quad 4 < 7$

This comes out of the rule but also by Logic if we use the quantized version.
But observe that the above explicit definition implies this:

$\forall x \forall z [x < z \leftrightarrow \exists y (x + y = z)]$

This could be used to derive cases too and could give different ones than we get by the above rule.
So we should have used different $<$ symbols for the explicit and the marked effective one until we prove that they are the same.

We could now define a next level for repeated exponentiations, but a much smarter thing is to define all possible $x\ [n]\ y = z$ relations as quadruples.
The $n = 1$ case is addition, $n = 2$ is multiplication, $n = 3$ is exponentiation and so on.
The general rules are easy!
The $x\ [1]\ 1 = z$ relation is simply $x \lhd z$ so this as condition is enough.
If $x\ [n]\ 1 = z$ with $n \neq 1$ then $x = z$ so this as condition is enough.
Finally, the crucial third line will say that to step in y from a previous v, we need not just the standing of $x\ [n]\ v = w$ but also the standing of $w\ [k]\ x = z$ for the previous k :

$$\left.\begin{array}{r} (n = 1)\ (y = 1)\ (x \lhd z) \\ (n \neq 1)\ (y = 1)\ (x = z) \\ (x\ [n]\ v = w)\ (v \lhd y)\ (k \lhd n)\ (w\ [k]\ x = z) \end{array}\right] \rightarrow \quad x\ [n]\ y = z$$

The added beauty is that this is independent of the earliers.
Multiplication used addition and exponentiation used multiplication as earlier targets.
To see the last crucial line even better, observe that indeed, to increase the multiplication value from a v second variable to the next y we just have to add x.
To increase the exponentiation value from v to y we must multiply by x.
And to increase the [4] value from v to y we must raise the old result to the x power.

So $x\ [4]\ 1$ is x by the second line and then $x\ [4]\ 2$ is x^x.

Then $x\ [4]\ 3$ is $(x^x)^x = x^{(x^2)}$ and we shouldn't write simply x^{x^x} for this because the $x^{(x^x)}$ value is different and could be understood for it just as well.
This is a consequence of exponentiation not being arbitrary in its order unlike addition and multiplication. So, for multiplication we can simply say that it is repeated addition and for exponentiation that it is repeated multiplication but for $x\ [4]\ y$ we shouldn't just say that it is repeated exponentiation.
A simpler fact is that the exchangeability of order already fails for exponentiation.

Indeed, $2^3 = 8$ but $3^2 = 9$ which feels trivial but same rules created addition and multiplication and for those the order is irrelevant. This is a mystery!
Observe something else! For the three normal operations z is always at least as big as x or y.
Now with $x\ [n]\ y = z$ this remains but the third input n can grow above z because for example, for all n values we get merely x as initial z value at $y = 1$.
This fact, that we can get small z values for big n hides the quite opposite fact how fast z grows with x and y. This caused big havoc at the early recursion period of Effectivity.

Then the normal operations were first generalized in a different direction as f functions that are given initial values and then defined for step by step increasing inputs.
These "primitive recursive" functions can not grow as fast as our x [n] y and this was regarded as a complexity of x [n] y = z.
In truth, this as relation is very "primitive" for the following reasons:
Our rules give the impression that the collectable tuples are complex because we have the freedom to choose derived cases and stupid choices can become an infinity of cases that are only a very small subset of all derivable cases. Indeed, at < for example, we can merely have the axioms $1 \lhd 2$, $2 \lhd 3$, $3 \lhd 4$, . . . as derived cases. But we can be much smarter too.
We can regard an m fixed value and try all numbers up to m as inputs that is case conditions in all our finite many defining conditions. Obviously as start we'll only be able to use these numbers in the $\lhd$ basic relations there. Then we get some target tuples and we again only regard the ones using values up to m. These can now be used in our second try of all conditions. We again get new tuples with values up to m and we use these again. Repeating this, we get a stage where no more new targets can be obtained. Simply because we couldn't get infinite many target tuples by using only numbers up to m. So we derived all target tuples with using values up to m.
Observe that this m will also be the maximal value in all of our derivable targets above.
Indeed, in every target in our rules above, the maximal variable value was always at least as big as other values in the conditions. Namely, the maximal was always z, except at x [n] y = z.
And here if n is the same or bigger than z then the only variable not up to z in the conditions is k but it is under n.
m appearing as maximal in the targets then implies that a target with values up to m can not come about by increasing m to an M. Simply because then M will be the maximal.
This means the same that the tuples not in our targets up to m will remain outside tuples as we increase m. So we can derive the outside tuples too as follows:
We derive the inside ones in the above systematic manner using m = 1 , 2 , 3 , . . and at every such m value once we have all the derivable ones, we check all tuples formable up to m and list the ones not derived.
This is a bad news because we aimed to find effective collections without effective complement.
A simple "solution" jumps to mind. Let's destroy this feature that our targets were so maximal.
We can simply omit some target variables and thus make them merely condition variables.
So now we allow u , v , . . as earlier target variables and then the rule is simply:

$$R (u , v , . . , x , y , . .)] \quad \rightarrow \quad T (x , y , . .)$$

Logic sense this means that from an earlier R target we obtained $\exists u \exists v$ . . R as T.
Can this be enough to get non effective complements? Or an other question is this:
Above we saw that our rules were such that the complements were all effective.
Are there deriving rules at least for these complements too?
And here we have again a very general solution. Namely, if an F (x , y , . . . , z) relation is functional for its last z variable, that is for any x , y , . . values z is unique then

$$F (x , y , . . . , w) , \; w \neq z] \quad \rightarrow \quad T (x , y , . . . , z)$$

will define a T target that is actually $\neg$F.
Observe that the condition of all possible other variable values was crucial!
Without this we would derive correct missing values for $\neg$F but not all of them.
Indeed, the missing x , y , . . values with any z would not be obtained at all.
If however F is a total function, defined for all x , y , . . variables then such missing tuples simply don't exist.
Now we should see two practical sets from Number Theory.
First the composite products can be at once defined by multiplication with added conditions:

$$x \cdot y = z \; , \; x \neq 1 \; , \; y \neq 1 \;] \quad \rightarrow \quad x \cdot y = z \text{ comp}$$

This is just like multiplication so its target contains the maximal tries and thus its complement is effective. It means $x \cdot y \neq z$ comp so dropping out the x , y variables would not give the primes and 1 rather all values because any z can be a false multiplication value.
Of course, we can drop out the x , y variables from $x \cdot y = z$ comp as:

$$x \cdot y = z \text{ comp }] \quad \rightarrow \quad \text{Comp } (z)$$

The previous failed attempt can be repeated by the false idea that:

$$\text{Comp } (w) , \; z \neq w] \quad \rightarrow \quad \neg \text{Comp } (z).$$

This would just give all z values again.
To derive the primes, we need to negate all compositions of z from numbers under z:

$$1 \cdot 1 \neq z \; , \; 1 \cdot 2 \neq z \; , \; . \; . \; . \; , \; (z-1) \cdot (z-1) \neq z \;] \quad \rightarrow \quad \text{Prime } (z)$$

So the dots that our rules tried to avoid, sneaked back!
Luckily, we can avoid these too with an other tricky rule.
Indeed, from an R we can obtain the $\forall \; u < x \; R$ relation as T target with the rule:

$$\left.\begin{array}{c} x = 1 \\ \\ T (u , y , \; . \; . \; . \; , z) \; , \; R (u , y , \; . \; . \; . \; , z) \; , \; u \lhd x \end{array}\right] \quad \rightarrow \quad T (x , y , \; . \; . \; . \; , z)$$

Using this first for $R = (u \cdot y \neq z)$ we get $T (x , y , z) = \forall u < x (u \cdot y \neq z)$, and then for this as R with y we get $T '(x , y , z) = \forall u < x \; \forall v < y \; (u \cdot v \neq z)$.
And thus $T '(z , z , z)$ means Prime (z) . The $x \cdot y \neq z$ must of course first be derived.
So the full set of rules for the primes is this:

$$\left.\begin{array}{c} (x \lhd z) \; (y = 1) \\ \\ (x + v = w) \; (v \lhd y) \; (w \lhd z) \end{array}\right] \quad \rightarrow \quad x + y = z$$

$$\left.\begin{array}{c} (x = z) \; (y = 1) \\ \\ (x \cdot v = w) \; (v \lhd y) \; (w + x = z) \end{array}\right] \quad \rightarrow \quad x \cdot y = z$$

$$x \cdot y = w , w \neq z \;] \quad \rightarrow \quad x \cdot y \neq z$$

$$\left.\begin{array}{c} x = 1 \\ \\ T (u , y , z) \; , \; u \cdot y \neq z \; , \; u \lhd x \end{array}\right] \quad \rightarrow \quad T (x , y , z)$$

$$\left.\begin{array}{c} y = 1 \\ \\ T '(x , v , \; z) \; , \; T (x , v , z) \; , \; v \lhd y \end{array}\right] \quad \rightarrow \quad T '(x , y , z)$$

$$T '(z , z , z)] \quad \rightarrow \quad \text{Prime } (z)$$

Alteration sequences

Returning to Turing's idea of alteration sequences, just like the finalized derivations of mathematics have a language, these mechanized alteration sequences should have one too.
So we need an other alphabet say the Greek, in which the normal Latin text alterations are texts again. This gives a new wild idea at once.
We list all our Greek texts as Ψ_1 , Ψ_2 , . . and regard again a non-continuing B_1 , B_2 , . . . Latin text list. If we can define the effective Latin text sets as $[\Psi]$ to any Ψ effective method, that is Greek text, then a simple translation of Ψ to Latin could be $|\Psi_n| = B_n$.
So then assigning $[B_n] = [|\Psi_n|] = [\Psi_n]$ is a perfect method as normal assignment.
Luckily we don't need this because a much better natural $|\Psi|$ translation comes about

Turing's heuristic idea of the effectivity steps were alterations of an A text letter by letter.
This is going back to school and I mean elementary school. The digital calculations of addition, subtraction, multiplication and division are done digit by digit and thus involve three things:
Reading a digit, writing a digit and moving to next.
Just like at reading a book we need a ruler if we are too young or too old, here too actually a finger or pointer or cursor is needed to tell where we are.
If the digital calculation rules we teach in elementary school can produce these four operations that then define functions for all numbers, then why don't we allow other digit by digit rules to produce all possible effective number functions or number collections. To use letters instead of merely digits is a big but formal widening. The more concrete widenings must come with narrowings. The big widening is that we are allowed to move to the next digit not just forward in an execution order but backwards too. The rules must tell at every step which way we move.
The big narrowing is that we can only move left or right not in two dimension as the elementary school rules do. The most crucial widening is that we don't have to write the new digits in empty squares rather we can write them in place of old ones too. So in elementary school terms this would need an eraser. This is where a machine is different from a paper because it can store in each square a digit or letter. It can be rewritten as many times we want it.
So the system is very simple. We have an infinite line of memory squares and we read write and move square by square. We can read m possible symbols of our alphabet and we can act in 2m possible ways by writing any letter in place of the old and then move left or right. There we read again. At the simplest binary alphabet, we have only 0 and 1 and four actions as:

$(0, \leftarrow)$, $(1, \leftarrow)$, $(0, \rightarrow)$, $(1, \rightarrow)$. Thus we have eight reactions as:

$0 \Rightarrow (0, \leftarrow)$, $0 \Rightarrow (1, \leftarrow)$, . . . , $1 \Rightarrow (1, \rightarrow)$.

An infinite sequence from these eight options can bring about any alteration of the data line.
This is not so surprising. In one direction we could simply go always forward and replace every digit to what we must bring about. This of course would leave the other infinite half unchanged.
Going back and forth we can alter everything to what we want.
This was interesting because if our task is to wipe out the memory line, that is write 0 to every square, then this trick of going back and forth is again needed but we don't need the infinite target line that we must bring about. So this should be an effective task and writing always 0 is easy but how could we bring about the back and forth motions. We can count the increasing same directional moves and shift direction at the right moments. Could we do this without external countings? The extremely simple solution is this. We don't go in equal motions. Our task is wiping out the 1-s so we only have to go until we see a 1. There write 0 and change direction.
Repeating this will alter all 1-s to 0 but only if we do have infinite many 1-s in both direction.
If we only have infinite many in one direction but finite in the other then this would go into a wild goose chase after a few returns because we always would encounter only 0-s.
The simple going forward and writing 0-s will do the job for one directional wiping out but we have to know where we start. This also works to wipe out finite many 1-s of course.

These raise big doubts about if simple effective programs could even achieve tasks without external knowledge. The other big question is how could we define a choice sequence from these eight reactions as effective. Amazingly, this is the easier.
Finite many of them obviously must be effective because we could regard them as a rule book.
But how could we organize them? Most logical is by the last few read symbols.
So imagine the rule decider as a person who only gets what was read but he writes these down.
After the start if he gets 0 or 1 then he writes this down and will look in the rule book for 0 or 1 as entries and give the order for one of the four possible actions. Then when he gets the next 0 or 1 he will continue the previous on his paper with this, so he can look up the rule book for a double digit say 01. He finds there again the action he must give out as order. At the tenth reading he will have already a fix nine long 110100010 number on his paper and will add the new tenth digits to this, look it up in the rule book and give the order.
Amazingly, for general alphabets the whole thing is even more heuristic.
We can have m many symbols to be read but these will form on the decider's paper a text.
This is the "time text" that the decider received as last read symbols and uses to decide what to do.
But of course he is a true bureaucrat, not making any decisions at all.
He only looks up these time texts as "entries" in a rule book that contains all possible reactions.
The order of the entries could be ordered in two ways. Alphabetically or length alphabetically.
We choose this second. The possible order to an entry must always contain first the action to be done on the line. This is the symbol to be written plus $\leftarrow$ or $\rightarrow$.
To make the rule book finite it must redirect some entries to earlier entries.
Earlier is now meant in length alphabetically because we chose this as order of the rule book.
These are given as return texts after the just mentioned ordered action on the line.
To make it even more practical, this redirecting shouldn't be given when the decider already gets the new symbol, rather when he was giving the last. So there is no unnecessary wait for action.
If "apple" is not continuable then in the rule book when they give the action for apple they also give "car", telling the decider to wipe clean his paper and write instead of "apple", "car".
So if the next incoming symbol is "s" he can continue as "cars" and look this up instantly.
If there is no return text given then this means merely to continue the time text readings.
Our agreement to make the returns in advance is very useful for making a rule book itself.
If we regard every text as Tx , that is its last symbol separated and call at a Tx entry, T the pre-entry, then in a rule book we always find all possible x symbols after T.
Indeed, the returns are giving these T pre-entries and all possible next symbols can happen.
The only drawback of this method is if a return is same length as the entry.
If for example at the "cider" entry we find the "apple" return then this is a correct one because "apple" is length alphabetically earlier to "cider". But at reading "cider" we are in the "cide" pre-entry which is shorter and thus earlier than the given "apple". So we will jump forward.
But observe that the "apple" pre-entry had been decided already in the "appl" pre-entry at the "e" continuation by not having a return there. So this single symbol forward jump is a consequence of our choice of making returns to pre-entries. But this allows all entries for the pre-entries.
And the advantage of this outweighs this inconvenience of the one symbol forward jumps.
An interesting case of this advantage is that even a return to the start can be easily interpreted.
We merely have to use the Δ empty text as pre-entry. So Δa , Δb , . . , Δz are always the first m many entries in a rule book. A return to the start is simply giving Δ as return pre-entry.
Then for later pre-entries we omit this Δ and only write the real texts.
Here is an example for the first line of a rule book and what it says.
$\Delta a \Rightarrow (c, \rightarrow)$ This says that if you see an "a" then write "c" instead and move right.
No returns were given which is not surprising in the first line.
Suppose we indeed started from an "a" on the line and so this first line was applied.
So we moved right and say there we see a "p". We must go to "ap" in our rule book and there will be such entry because at "Δa" we didn't order return so "Δa" = "a" will be a pre-entry.
So we will have all "ax" texts as entries and at "ap" we may find this:
$ap \Rightarrow (i, \rightarrow)$
So replace p with i , move right and again no return is given so "ap" will be a pre-entry.

So again we have all “apx” entries.
If we see a “p” again on the line, we can have in our rule book:
app $\Rightarrow$ (d , $\rightarrow$)
Say we see now l on the line and we find in our rule book:
appl $\Rightarrow$ (e , $\rightarrow$)
Finally we see e on the line and our rule book says:
apple $\Rightarrow$ (r , $\rightarrow$)
We rewrote apple to cider on the line.
But this was a very unusual situation, not just because no returns were given at any of our entries but because we kept going right. Normally, the Ψ (apple) time text would not come from “apple” on the line. Or that being on the line would not become a time text.
Starting from the left end of “apple” means nothing for the machine or the rule book.
We might at once get the order to move left and see “pple” never again.
So to make sense of a T alteration sequence requires again some non-continuing assumptions.
We should have that T is the widest in some sense. On an infinite line of course this is not obvious. One solution could be to regard blank memory cells of the full line and only allow finite starting text but then we would have to allow writing blanks that is erasing. The simplest is to allow blank as symbol in our alphabet or in effect, start with a line where all cells from a point both left and right are same symbols which is thus regarded as the blank.
With the binary alphabet we should have all 0-s from a point both left and right.
Then the T text is the digits between the two first of these infinite 0-s.
In other words, our text is the text from the first 1 to the last 1 including these.
Then it is meaningful to regard T Ψ as two infinite alteration sequences.
One on the line in Latin and one in our rule system in Greek:

$$T\Psi = \begin{matrix} T_1 , T_2 , . . = (T\Psi)_1 , (T\Psi)_2 , . . \\ \Psi_1 , \Psi_2 , . . = (T\Psi)^1 , (T\Psi)^2 , . . \end{matrix}$$

While the upper T texts can grow arbitrary in length, the Ψ texts have limited lengths.
The finite Greek rule system is like Physics since Newton. Laws of matter. Applying it to the universe, we need external conditions. These are the initial T texts we start from.
Now comes the “heart of the matter” idea.
We allow a stop in the rule book’s T entry line with a return to the T’ pre-entry, by not doing this return, that is “halting” it. So in effect, T’ becomes irrelevant and can be replaced by “halt”.
Without halt condition every Ψ runs forever for sure and with halt condition it may very well too.
Simply because the T entry line that contains halt will not be encountered.
If Ψ_n contains halt then the corresponding T_n is regarded as the final alteration result from T.
This would make us think that we’ll collect as [Ψ] these $R = T_n$ results for all T inputs where such exists. But a much simpler idea is to collect the T inputs themselves.

[Ψ] = { T ; for some n , Ψ_n contains halt } = in short as: { T ; T Ψ halts }

$\neg$[Ψ] = { T ; there is no n that Ψ_n contains halt } = { T ; T Ψ runs forever }

So we totally ignore the R result in our collections. These results will become only interesting if by chance Ψ is such that it halts from any T and thus the R result is an effective transform.
The whole point of Turing’s system is that it grasps those effective collections that are beyond this narrow concept of effective transform. Indeed, as we mentioned, some recognitions, that is [Ψ] collections by halting inputs are such that the $\neg$[Ψ] complement set of texts, that is the inputs from which Ψ doesn’t halt, are not collectable by any process at all.
So then [Ψ] is definitely not replaceable by a collection of always existing results merely selected by some decision about the results being such or such.

Next we introduce an effective non-continuing $|\Psi|$ translation of all Greek texts to Latin.
This is unique and effective in both directions. So from a $|\Psi|$ translate we can recover the Ψ machine. Most importantly a universal Ω machine can not just recover Ψ from $|\Psi|$ but simulate it too. Namely, by using the $|\Psi|$ translate as beginning added in front of any T text.
So the alteration sequence $T\Psi$ is simulated by $|\Psi|T\Omega$.
The success of simulation is of course: $T\Psi$ halts $\leftrightarrow$ $|\Psi|T\Omega$ halts.
This must imply for Ψ used as Ω itself that:
$T\Omega$ halts $\leftrightarrow$ $|\Omega|T\Omega$ halts $\leftrightarrow$ $|\Omega||\Omega|T\Omega$ halts $\leftrightarrow$ . . .
Remember how we planned to define our assignments:
$[A]$ = empty if there is no Ψ that $A = |\Psi|$ while $[A] = [\Psi]$ if $A = |\Psi|$. So:

$[A] = \{ T$; for some Ψ, $A = |\Psi| \wedge T\Psi$ halts $\} = \{ T$; for some Ψ, $A = |\Psi| \wedge |\Psi|T\Omega$ halts $\} =$
$\{ T$; for some Ψ, $A = |\Psi| \wedge AT\Omega$ halts $\}$. In particular:

$[|\Omega|] = [\Omega] = \{ T$; $T\Omega$ halts $\} = \{ T$; $|\Omega|T\Omega$ halts $\}$ as expected already.

Since the $|\Psi|$ translates are non-continuing, we can define our universal set by simply combining all $[\Psi]$ sets with the added $|\Psi|$-s as beginnings to all texts:
$\mathcal{U} = \{ BE$; for some Ψ, $B = |\Psi| \wedge BE\Omega$ halts $\} = \{ T$; $T\Omega$ halts and T has translate beginning$\}$.
This will be universal because every $[A]$ can be obtained from $\mathcal{U}$ as $\mathcal{U} - A$.
Indeed, if $A = |\Psi|$ then $\mathcal{U} - A = [A] = [\Psi]$ so we get every assigned set.
If A is not a $|\Psi|$ then $[A]$ is empty but $\mathcal{U} - A$ not at all.
A maybe shorter or longer than a $|\Psi|$ so we cut off too little or too much to get $[\Psi]$ but we definitely won't get an empty set.
We could define our assignment as $[A] = \mathcal{U} - A$ for all A texts but it wouldn't make much sense. The whole point is to carry back the universality of Ω to the Latin texts.
For $A = |\Omega|$ itself the universality of $\mathcal{U}$ implies that $\mathcal{U} - |\Omega| = [|\Omega|]$.
But to have $|\Omega|$ as name of $\mathcal{U}$ that is $\mathcal{U} - |\Omega| = \mathcal{U}$, we must have $\mathcal{U} = [|\Omega|] = [\Omega]$.
This requires that Ω halts only from texts that have translate beginnings.
But it must ignore added $|\Omega|$ beginnings as we saw above.
Thus we get what we already expected by $\mathcal{U} - |\Omega| = \mathcal{U}$, namely that for any T inside $\mathcal{U}$, $|\Omega|T$, $|\Omega||\Omega|T$, . . . are inside too.
By our agreement to make the non-proper assignments empty, $[|\Omega||\Omega|]$, $[|\Omega||\Omega||\Omega|]$, . . are empty too. But we shouldn't jump to conclude that thus $|\Omega|$ is the only name for $\mathcal{U}$.
Indeed, we should be able to alter Ω to an Ω' so that it doesn't affect its alteration of T texts.
So $[\Omega] = [\Omega']$ and thus $|\Omega|$ is only one of the magic texts and it is still only abstract anyway.

Wandering away

The most obvious way of getting a Ω' variant of Ω is to add irrelevant steps to Ω.
This same works for any Ψ to get Ψ' variants. But the really hidden feature of machines is that they bring about variants by themselves that we can not even see from their rule books.
At real machines, even at computers, we can get a good price for superficially damaged items that otherwise work "perfectly". So, there is a perfect functioning that is unique. At machines as representations of Effectivity we have an infinity of perfect variants. The only way we could find "bests" among them by regarding the shortest ones. As I mentioned in the very short section, this was an important point to define the complexity of texts. But now these variants are important for very different reasons. This relates actually more to the opposite feature, redundancy.
The twenty amino acids are coded by four valued triplets of nucleic acids. That means 64 possible codes, so nature made a huge redundancy. But even in our results at the start we had hidden redundancies that we simply ignored. Remember the decimals that had the extra forms.
But even earlier when we sequenced the fractions as:

$$\frac{1}{2}\ ,\ \frac{1}{3}\ ,\ \frac{2}{3}\ ,\ \frac{1}{4}\ ,\ \frac{2}{4}\ ,\ \frac{3}{4}\ ,\ \frac{1}{5}\ ,\ .\ .\ .\ .$$

We actually we wasted places here because we repeated fractions like $\frac{2}{4} = \frac{1}{2}$.
To leave these out is of course easy but would be an extra ugliness.
But that I am not bullshitting, let's try to list all fractions not just the ones in [0,1].
The trick is the same, that is going in finite groups but the fix denominators wouldn't work now.
Indeed, now there are infinite many fractions with any fix denominator.
The solution is to go in the totals of the numerator and the denominator:

$$\frac{0}{1}\ ,\ \frac{0}{2},\frac{1}{1}\ ,\ \frac{0}{3},\frac{1}{2},\frac{2}{1}\ ,\ \frac{0}{4},\frac{1}{3},\frac{2}{2},\frac{3}{1}\ ,\ \frac{0}{5}\ .\ .\ .\ .$$

Here we have a "redundancy galore" and strangely we listed much more than previously.
So it seems the more you squeeze into a smaller place the more you have to be generous and give multiples as presents.
Now back to our machine variants, the really important hidden fact behind their features is why we could recognize these features at all. Because we used the right concept of "variant".
In fact, I still didn't reveal this and only used it above vaguely.
But from our machines as text alterators, the exact definition is almost trivial.
Two Ψ and Ψ' machines are variants if they collect the same texts, that is if they halt exactly from same inputs. So this trivial definition of variants that reveals the inner secrets of machines is only possible because we did the initially so stupid direction of following the same seemingly stupid definition of sets being mere collections. As I said then, the end justifies this start.
So now we can see already that this variant concept is a major part in that end. But to formally declare machines as text collectors could have not been enough. Turing's heuristic text collecting idea by alteration sequences halted by a purely inner condition of the machines, was the method that made this text collecting idea universal. And "universal" here meant externally universal, that is grasping all effective collections. This is what we can not prove only experience.
The inner universality of Ω is provable but this is merely a tool to see features of machines that are universal externally again. Turing didn't realize this and attached too much importance to his universal machine. This is understandable because this was the only origin of effective collection without effective complement.
Kleene was the first who peeked into the surprising world of variants. His Recursion or Fix Point Theorems hit upon a crucial feature. The fix point name refers to what the theorems really say while the recursion name to an amazing use of the theorems. Namely, that every recursive rule,

like the one we used to define x [n] y = z can be melted into the non “free” systems easily if the operations inside conditions are there. So the recursion itself is automatically inside.
This was a huge step in seeing the external universality of these totally mechanical effectivities.
But just as Turing was attached to the universal machine, Kleene was to the fix point.
The third step which I regard the breakthrough in Effectivity, was to see that variantness can be an obvious condition if we collect translates, that is programs by how they run.
Then the non effective complements become an almost necessary situation.
So all we need is to use all texts, that is all data, as potentially runnable program. Not merely alterable dead data. Then we will get really complex collections without effective complements.
So the previous two stages, inner universality and fix points are merely the subjective steps to derive this final and robust experimental, almost physical truth.
It is derivable in any correct framework only because it is a truth as such. And not in reverse.
So Platonism or Objective Idealism had been beautifully exemplified in this progress.
The important role of subjectivity shows that this is the correct way to see behind subjectivity too.
The unpredictability of subjectivity stands! But behind every unpredictable truth of subjectivity stands a bigger objective truth. The stage of subjectivity when we still don’t see the bigger truth is unavoidable. It is Life! But the hollow relativism that denies objective truth is denying everything. Subjectivity too! These are the funny people. And I mean always funny. Who can never turn serious and make a judgment. These just smile at the world’s stupidity and don’t want to point fingers or name names. After all, who has the right to criticize others. We are all sinners, consumers and weak. So under the big blanket of this relativism can grow the new stupidity of the petty bourgeoisie. The social status quo of relativism is the new age. Everybody is free to think whatever he wants. The manipulators are busy cashing in on this gold mine of “freedom”.
The fascist petty bourgeoisie was only a temporary stage. So Dimitrov was very deep recognizing something but it’s over. There is no need for a single megalomaniac to bring about the alliance of the petty bourgeoisie and the oligarchs. Now the whole world as such is the negotiator of stupid thoughts and therefore actions. What we think is what we are. What we do is just options from the tray of the evil negotiator. The spreading New Myth is that there are “Some” who organize this whole system. Conspiracy theories are merely wild branches of this new myth. That people create this framework that people live in. The “Some” are coming together and decide War and Peace.
But even Tolstoy missed the essential counter truth. How individual actions, seeking a deeper freedom fall victim to themselves. Actions without consequences die off like ripples caused by pebbles. The broken glasses get thrown out and the window is repaired. Naked fans on the field can even make the headlines. But the mad men who want to change history must pass some tests. Must be excellent sharp shooters or must be able to take off a plane, even if not make it land.
But then we might say what is a change in history anyway. What would be different if Kennedy lives or the trade center stands?
And that is so because we are in the new age and indeed a certain history has been killed.
The last two mad men who got to the top were Stalin and Hitler. And Hitler won! By losing.
Stalin didn’t alter history and Hitler altered everything!
My dreams about an alternate history where Hitler won by winning and blamed everything on Stalin, were so rich in details that I have no doubt they have a corresponding objectivity behind.
The old Hitler gives a speech in New Delhi as UN secretary to implement the Resource Act, that changes everything about nations. The Auschwitz Group is a small circle of anarchists who obtained evidence about Hitler’s involvement and try to unmask the Old Man.
This is actually a preferred name Hitler likes to be called. The speech is about ten thousand words and I can remember every one. The taste of this future is still in my mouth and will be there forever. So much better than ours in many respect and I hate it just as much as this one. The one I think I live in. So I don’t even try to dream a future willfully. Probably I would hate that too.

Busy beavers

To get a better feel of how simple yet how complex Turing's machines are, we should see some examples. To make this useful in an other sense too, we should realize something amazingly simple yet quite unbelievable. I forgot this fact three times and every time I realized it again I was astonished. It's about the so simply defined halt condition.
Every rule book cuts off the needed memory writing of the decider by the returns to shorter ones.
So if we regard a given fix maximal M as the decider's memory length, that is time text length, then we can ask any things about how these limited machines behave.
The most important question is if they halt or not. But this of course depends on the T start.
Allowing a given say M = 10 time text limit and say 100 as T length limit, the possibilities are astronomical. So we feel that the possible halts or non halts will be two infinitely observable distinctions and thus infinite many longer and longer runnings can be brought about.
Amazingly this is false! There is a step number after which halt is impossible! And the explanation is the mentioned simple logic that I forgot three times.
The given M = 10 limit on the time texts means rule books with maximum 10 long entries.
There are only finite many such and the possible 100 long starts are again a finite number.
So T Ψ has finite many variety. Checking each empirically we will see if one halts or not.
This means a maximally running one among them and so beyond that many steps none will halt any more. The second, more empirical surprise is how astronomical the possibilities are already coming from M with even 0 length limit on T. This shouldn't be written as T = 0 because it may mean a 1 length start if 0 is used as data. Of course, with 0 blanks they mean the same.
But to be more theoretical, we could use again Δ just as we used it in our rule book.
So T = Δ can mean the empty entry, that is blank entry as starting text in general.

Tibor Rado realized how complex these running condition can be even with Δ start, that is from the all 0 line with a binary alphabet and with very small M rule book entry lengths.
At M = 3 the maximal step number is already 21.
Most important is that these behaviors are unpredictable, so only empirical examinations can tell how the possible machines will run. But to check all possible machines is a huge task.
To find the maximal step number is thus the best thing because then we know how long to check the machines for other behaviors among the halting ones.
For example, he asked first how many 1-s can be produced by halting machines.
Interestingly, the maximal 21 step machine will produce five 1-s but a much shorter 13 step machine can produce six. So the longest running is not necessarily the most in every respect.
If we check all M = 3 machines up to 21 step, we find this one too that is better in this respect.
The busy beaver name refers to how these machines run back and forth.
So here is the rule book of the M = 3 busy beaver champion for step number.

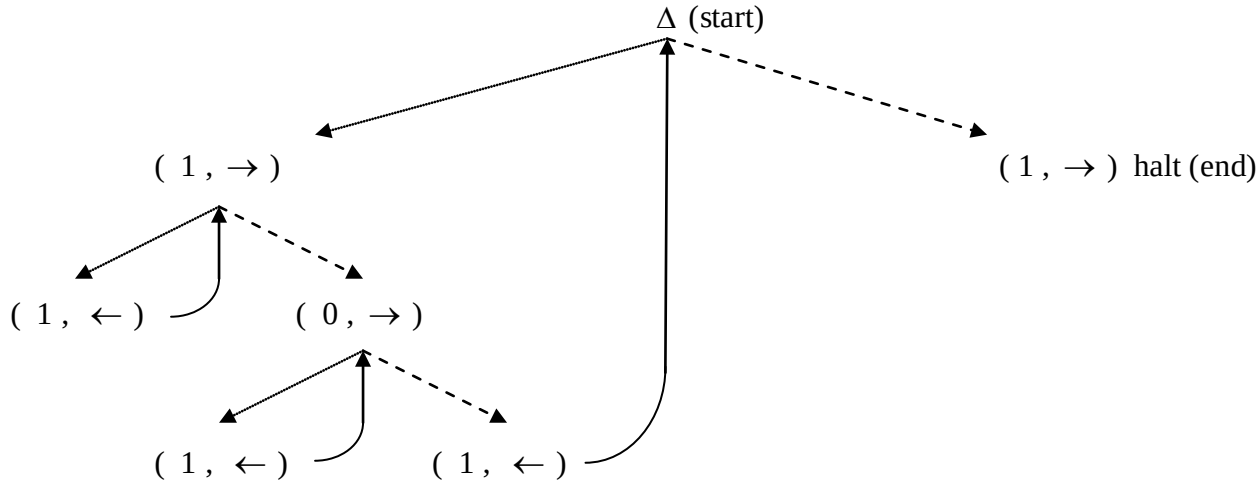
0 ⇒ (1 , →)
Δ
1 ⇒ (1 , →) halt

0 ⇒ (1 , ←) return to 0
0
1 ⇒ (0 , →)

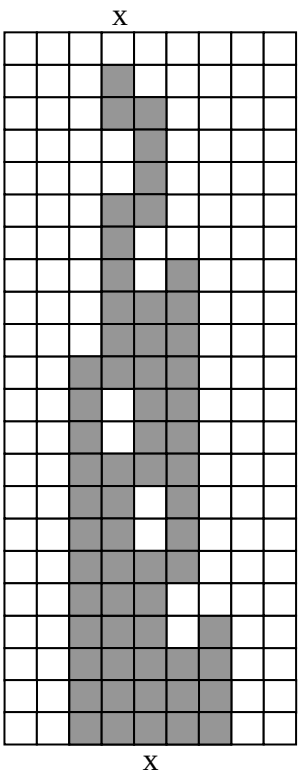
0 ⇒ (1 , ←) return to 01
01
1 ⇒ (1 , ←) return to Δ

The entries that have return pre-entries in an M rule book are a non-continuing set of texts with at least an M maximal length one among them, so that every M long text is either such or is a continuation of an element. Cutting off the continuations from these texts in the full text tree, we

are left with a “finite pruned tree”. The junction points don’t have to be marked by texts or even symbols because their position tells those exactly. So instead, we can place the actions ordered by the rule book to that text. Under the chopped off dead ends, in addition we can write the return pre-entries or even use arrows leading to those points in the finite pruned tree.
Thus, these arrows must always go to length alphabetically earlier branching points.
Such redirected finite pruned tree with the actions at every point, could be called an “action tree” and it is the best visualization of a rule book. It is a special version of the “flow charts”.
Here is the action tree version of the above rule book for the M = 3 busy beaver champion for step number:



Here is the actual action on the line with whites as 0 and greys as 1.
The x marks the start and end position of the reading head.



Turing didn’t realize the “time text” as a heuristic organizer of the entries. And indeed, we can just say Δ ~ state 1 and continuingly the pre entries as states 2 , 3 , . . . , n , . . . , N.

In our case: $0 \sim 2$, $01 \sim 3$. By coincidence, we get again 3 so our 3 length limitation corresponds to the same $N = 3$ state number limitation.
Without the time texts of course, the natural continuations are meaningless and indeed, Turing marked all state jumps. So with state notation, our rule book looks like this state table:

$1 , 0 \quad \Rightarrow \quad (1 , \rightarrow) , 2$
$1 , 1 \quad \Rightarrow \quad (1 , \rightarrow) ,$ halt
$2 , 0 \quad \Rightarrow \quad (1 , \leftarrow) , 2$
$2 , 1 \quad \Rightarrow \quad (0 , \rightarrow) , 3$
$3 , 0 \quad \Rightarrow \quad (1 , \leftarrow) , 3$
$3 , 1 \quad \Rightarrow \quad (1 , \leftarrow) , 1$

The reverse, that is finding a rule book for any state table is more difficult because with blind states we can prescribe any jumps ahead not just to the continuing pre-entries or the mentioned one symbol forward jumps.
This is an advantage of state tables that they can not be “wrong” in their jumps.
We on the other hand may have a false return text that is not a pre-entry.
Still, to make a rule book for a state table is always possible by going through all texts in length alphabetical order. We’ll decide which ones are kept as branchings in the text tree and assign states to these. The actions themselves will be ordered as same by these assignments.
The start is keeping Δ for sure and assign to it state 1 : $\Delta \sim 1$.
At the $a , b , . . . , z$ junctions in the tree, or at the $\Delta a , \Delta b , . . . , \Delta z$ entries in our rule book, we place the same $(\quad)$ actions found in the state table at:
$1 , a \Rightarrow (\quad) , n_a \quad , \quad 1 , b \Rightarrow (\quad) , n_b \quad , \quad . . . . , \quad 1 , z \Rightarrow (\quad) , n_z$
If n_a is halt or it is 1 then we place also halt or a return to Δ in our rule book.
This of course will mean that “a” will not be a pre-entry.
If n_a is a new state then we will make “a” a pre-entry and assign n_a to it: $a \sim n_a$.
If n_b is halt or it is 1 or n_a then we place halt or a return to Δ or a in our rule book.
If n_b is a new state then we will make “b” a pre-entry and assign n_b to it: $b \sim n_b$.
And so on, we always make returns to texts that already had been assigned to the states we find as jumps. Or if they are new, we assign them to the pre-entries we are going to make. Formally:
If we are in the T pre-entry at symbol x in our rule book or in the text tree under the T branching point at branch x, then to this Tx text or point we place the same action found in the state table at $n , x \Rightarrow (\quad) , n’$. Furthermore, if $n’$ is a state already assigned to a $T’$ and thus $T’ < T$ length alphabetically, then we simply make a return to $T’$. If $n’$ is not assigned yet, we assign it to Tx and make this later a pre-entry.
Eventually all states will be assigned in an $(M – 1)$-th line of the text tree and so the next M-th line will only have returns and so no pre-entries will be formed. In other words, all branchings will be copped off in the M-th line. We’ll have same many kept branching points, that is pre-entries, as states are in the state table.

Bigger pictures

For our effective assignments, 3. of the Basic Theorems implies at once that we must have some effective collection so that its complement is not effective that is not an assigned set.
Indeed, here a. is true as we saw, though only abstractly. Also b. should be true though we didn't check it exactly. So c. must be false.
Let's try to go back to the source, that is to find a Ψ_0 machine that has no complement machine, that is one that halts exactly from those starts from which Ψ_0 runs forever.
The idea is the same. We can say that two machines Ψ_1, Ψ_2 share a T text if either they both halt from T or both run forever, which means that $T\Psi_1$ halts $\leftrightarrow$ $T\Psi_2$ halts.
The only machine that shares no text with a Ψ_0 can be its complementing one which halts exactly from where Ψ_0 runs forever. So a Ψ_0 has no complementing machine if and only if it shares text with every Ψ. To find such Ψ_0 we again involve the universal Ω machine.
We talked about the effective transforms as the very narrow effectivity that halts from every T start and gives the R resulting text. To duplicate a T text into TT is obviously such task and so let Θ denote this. If we continue this with Ω then let's denote the obtained machine as $\Theta\Omega$.
We claim that this machine shares with every Ψ machine the $|\Psi|$ text.
Indeed, $|\Psi|\Psi$ halts $\leftrightarrow$ $|\Psi||\Psi|\Omega$ halts $\leftrightarrow$ $|\Psi|\Theta\Omega$ halts.
The doubling reminds us the Cantor Diagonality.
The anti-diagonal alteration we didn't need because the claim itself contained the complement.
In order to prove that a Ψ_0 has no complement machine, we don't have to always find text for every Ψ that would be a shared text with Ψ_0. For example, we can at once realize that Ω is without complement too. Indeed, if Ψ_0 were that then $T\Omega$ runs forever $\leftrightarrow$ $T\Psi_0$ halts. But then regarding for any A the AA duplicate as T we would get that:
$A\Theta\Omega$ runs forever $\leftrightarrow$ $AA\Omega$ runs forever $\leftrightarrow$ $AA\Psi_0$ halts $\leftrightarrow$ $A\Theta\Psi_0$ halts and so $\Theta\Psi_0$ would become a complement of $\Theta\Omega$ which we showed not to exist.
These two simplest impossible complement cases became called the halting problems.
Ω, the universal machine, indeed can be called the halting machine because it halts from those $|\Psi|T$ pairs where $T\Psi$ halts. $\Theta\Omega$ is then the diagonal halting machine. It halts from those texts that when doubled, Ω halts from that. For translates as texts, it means halting from those that correspond to machines that halt from their own translate as start.
If instead of doubling, we fix a T_0 start and regard all those Ψ machines that halt from T_0, then these $|\Psi|$ translates are an effective set. Indeed, Ω will recognize them because:
$T_0\Psi$ halts $\leftrightarrow$ $|\Psi|T_0\Omega$ halts. But will again be true that there is no complement?
Yes, and this is just a special case of a whole arsenal of non existing complements that were produced by a theorem discovered only in 1953. This theorem was not so hard to prove but its meaning changed our whole vision about effectivity.
Since the insane Formalist treatment of mathematics tries to ignore visions and only deals with theorems, this crucial theorem is only regarded as one in the line of others. In text books it's crucial meaning remains hidden. Of course, no text book can avoid mentioning it.
So to prove my point is very easy! Let's go back to the time before 1953.
Many regard it as the golden era of effectivity but I just call it the lingering darkness.

In 1931 Gödel started everything, by showing that in most axiom systems there have to be undecidable statements. From his proof it feels as if the language of these axiom systems were the reason for this. They can talk about their own derivabilities.

In 1936 Turing shows that effectivity can be defined naturally and sees clearly that derivability in an axiom system is merely a special effective collection of the theorems. The complement set of the theorems, the "non theorems" is not effective, that is not collectable by any method.

This was a first light in the tunnel. Indeed, with this, the existence of undecidable statements is not a language problem but has an almost physical cause:
If all S statements were decidable, that is S or $\neg S$ were theorem, then to collect the non-theorems effectively were possible as follows. We collect the theorems by deriving them all and then formally negate every one of them. The impossibility of any effective collection method for this set forces that the only loose chain in the assumption, the full decidability must be false.
Of course, we have same heavy guns as non loose chains here too:
Most importantly, how can we assume to derive all theorems? We have a time machine?
But this is merely a systematic application of all our axioms and all our derivability rules.
As start, the axioms if they are infinite many must be generable by some rules too.
This is true for all axiom systems though nobody noticed its importance before.
The rules of logic is actually the rule of derivations and these were quite exact by this time too.
So we can list all the theorems that all future mathematicians haven't even discovered yet.
As I explained, this is not paradoxical at all. This mechanical listing of all theorems is useless for a "real mathematician". I mean an old-fashioned one who regards mathematics as chess.
But for some, like me, this new math is the only math. To be most correct, we should call this new math as metamathematics and so Kleene for his first comprehensive book found the perfect title.
Once Paul Erdös said "it's time to do some real mathematics" after dwelling on some set theoretical problem. He was probably the last big classical mathematician. Stubborn too in his lack of vision for two things. That this new math is the only future math and that this metamathematics is actually relating to the understandability of mathematics by everyone.
So Didactical Logic is the real future. Unfortunately, it seems I am the only one believing in this.

In 1952 Kleene publishes his Bible the Introduction To Metamathematics.
And he still doesn't realize what I call the turning point. The seemingly so obvious theorem that now must be in every textbook somehow was forgotten to be included in the Bible by Kleene.
And this all relates to the always returning problem of non-effective complement!
The fundamental fact that universality implies their existence is crucial. But it can only create seemingly rare cases by always using the same diagonal trick with minor alterations.
Now we'll see that for some effective collections there is never effective complement.
It is not only a treasure chest of practical examples but explains why we were blind to them before. As start, we regard the already raised problem. Fix a T_0 start and collect all translates that correspond to machines that halt from T_0. We claim that the complement text set that thus must contain all translates of machines running forever from T_0 is not effective.
So there is no € machine that would halt from exactly these texts.
The basic idea of refuting such € machine is that it would imply that every machine has a complement. So observe a strange symmetry!
We claim that if an € machine could collect $\{ |\Psi| ; T_0 \Psi$ runs forever $\}$ for a fix T_0 then for any fix Ψ_0 the $\{ T ; T \Psi_0$ runs forever $\}$ set would be machine collectable too.
The proof however doesn't use this symmetry, rather goes by observing two trivial facts.
The first is that this assumed € machine would have to halt from all those $|\Psi|$ starts that are translates of Ψ machines that run forever from any T start.
Indeed, if a machine runs forever from any start it will do the same for the particular T_0 too.
The particular T_0 will not be thrown out though, rather used in the second triviality.
Namely, that some machines do halt from T_0 and actually all these must have translates from which € must not halt. In particular if a Ψ_0 halts from T_0 and Ψ_0' is a variant of Ψ_0 that halts exactly from those starts as Ψ_0 then € must not halt from $|\Psi_0'|$ either.
Now we can regard the arbitrary Ψ machine and create one that halts from its complement.
We "continue" this Ψ machine with Ψ_0. Meaning by this that if from a T start Ψ halts then we start Ψ_0 from a second T' start: $T \Psi \rightarrow T' \Psi_0$. For the translates this is a $T|\Psi| \rightarrow T'|\Psi_0|$

combined translate. $T\Psi \to T'\Psi_0$ will run forever for all those T that Ψ does and will run forever for those T' too that Ψ_0 does. For those T that Ψ halts, $T\Psi \to T'\Psi_0$ is merely a variant machine of Ψ_0. Thus € will not halt from $T|\Psi| \to T'|\Psi_0|$ for these T.

But it will halt for all T that Ψ runs forever because then $T\Psi \to T'\Psi_0$ is a never halting empty machine in its T' input.

So formally, $\{ T ; (T|\Psi| \to T'|\Psi_0|) \in \text{halts} \}$ would be the effective collection of $\neg[\Psi]$.

The tacitly used switcheroo of inputs and translates are the messy details to be proved.

We'll get a cleaner proof anyway though.

What is important is that we used much wider assumptions than the original plan was, collecting translates running forever from a fix T_0 start.

So not only such collections are not machine collectable, but any € emptiness "container" is impossible as machine. To explain this, we should define first "container" as text set.

An $\mathcal{S}$ collection of such and such texts is exactly the texts that are such and such.

An $\mathcal{S}$ container should be only a set that contains those but may have other texts inside too.

This of course, raises the triviality that the set of all texts contains any such and such texts.

What's worse, this full set of texts is machine collectable.

To exclude this triviality and other less trivial versions of it, we require from a container of such and such texts that it not only contains those but does not contain some $|\Psi_0|$ translate and neither any $|\Psi_0'|$ ones with Ψ_0' being a variant of Ψ_0. This means that Ψ_0 and Ψ_0' collect the same texts, that is halt from same texts, that is $[\Psi_0] = [\Psi_0']$.

An $\mathcal{S}$ emptiness container is then a container of all $|\Psi|$ translates with Ψ collecting nothing, that is halting from no start at all.

So now we can say meaningfully that there is no € machine that collects such $\mathcal{S}$ emptiness container. So $\mathcal{S} = [\in]$ is impossible.

To repeat again the earliers, the set of those translates of machines that run forever from a T_0 start, are an emptiness container too.

Indeed, the empty machines are among those that run forever from T_0.

And the required outside variants are satisfied too, because the set of all machines that halt from T_0 contain the variants too.

The original special case is already an infinity of non effective complements and a particular example in computer terminology is this:

The programs that stop from a fix input number say 100 are machine recognizable by simply watching any program run and recognize 100 to recognize such program itself.

But the complement, that is those programs that won't stop from 100, are not machine recognizable. So no program can stop from these and only these programs as inputs.

This more practical version as usual, brings out a deeper philosophical problem.

Does this no possibility of exact collection by halts mean that any analyzing of the programs is unable in general to detect the infinite running from 100?

Obviously for some particular programs we can establish by some "analysis" that they will run forever from 100. So the crucial point is that such analysis always must be particular. But this just opens newer questions that we can not answer because the concept of analyzing is vague.

To get the already praised theorem discovered in 1953, we again step back from the general impossibility of € machines, but not as much as we did for our initial case.

Here the particularness of the empty machines will disappear and we still get a much wider variety of non effective complements.

Instead of the requirement that we had a single $|\Psi_0|$ with variants outside $\mathfrak{S}$, we can require that there is such but also all translates outside have their variants there too. Such $\mathfrak{S}$ could be called a complete emptiness container. Indeed, observe the trivial fact that if the outside of $\mathfrak{S}$ is variant complete, that is contains all variant translates, then $\mathfrak{S}$ itself is variant complete too.
And this is true for any variant complete set. This then allows to avoid the emptiness container as condition altogether:

Rice's Theorem:

If $\mathfrak{S}$ is a variant complete text set and neither $\mathfrak{S}$ nor $\neg\mathfrak{S}$ are empty then both can not be effective. Only one or neither.

Indeed, they are both variant complete and one must contain all emptiness translates.
This then is a complete emptiness container that can not be effective.

We didn't talk about translates just texts because the variants are meaningful anyway.
If the two sets are variant complete then the non translate texts must be in only one of them.
Namely, in the one that is the complete emptiness container because the non translates are regarded as empty collections.
Now we'll get Rice's Theorem in a much more abstract way through two famous theorems of Kleene. The first is the Parameter Theorem and it is quite natural. It says that if Ω simulates a Ψ_1 with an input which is $|\Psi_2|T$ and so by altering $|\Psi_1||\Psi_2|T$ then there is a $(|\Psi_1|+|\Psi_2|)$ transform of $|\Psi_1||\Psi_2|$ so that it represents a machine that uses only T.
So the machine that has $(|\Psi_1|+|\Psi_2|)$ as translate has $|\Psi_2|$ built into it as parameter already.
Then of course Ω can simulate again, by altering $(|\Psi_1|+|\Psi_2|)T$.
The second is quite oppositely a very surprising result because it says that for every Ψ machine there is a Ψ_0 that Ψ shares all the $|\Psi_0|T$ texts with Ω. An even more surprising meaning of this is that Ψ simulates Ψ_0 just as Ω does, that is by beginning with its translate.
We'll prove the first version but use the second for Rice's Theorem.

Parameter Theorem:

There is a $(|\Psi_1|+|\Psi_2|)$ translate transformed from the $|\Psi_1||\Psi_2|$ text so that:
$|\Psi_1||\Psi_2|T\,\Omega$ halts $\leftrightarrow$ $(|\Psi_1|+|\Psi_2|)T\,\Omega$ halts.

Recursion or Fix Point Theorem:

For every Ψ there is a $|\Psi_0|$ that Ψ and Ω share all the $|\Psi_0|T$ texts. In other words:
$|\Psi_0|T\,\Psi$ halts $\leftrightarrow$ $|\Psi_0|T\,\Omega$ halts $\leftrightarrow$ $T\,\Psi_0$ halts. Thus $[\Psi] - |\Psi_0| = [\Psi_0]$.

Let's extend Ψ to Ψ^+ that first transforms B into $(|B|+|B|)$ for any BT start with translate B:
$(|B|+|B|)T\,\Psi$ halts $\leftrightarrow$ $BT\,\Psi^+$ halts $\leftrightarrow$ $|\Psi^+|BT\,\Omega$ halts.

Now let $|\Psi_0| = (|\Psi^+|+|\Psi^+|)$. Then indeed: $|\Psi_0|T\,\Psi$ halts =
$(|\Psi^+|+|\Psi^+|)T\,\Psi$ halts $\leftrightarrow$ $|\Psi^+||\Psi^+|T\,\Omega$ halts $\leftrightarrow$ $(|\Psi^+|+|\Psi^+|)T\,\Omega$ halts $= |\Psi_0|T\,\Omega$ halts

The first $\leftrightarrow$ is true by our definition of Ψ^+ with B used as Ψ^+.
The second is true by the Parameter Theorem with $\Psi_1 = \Psi_2 = \Psi^+$.

The first version that Ω shares so many texts with every Ψ is amazing but directly still just makes Ω a machine without complement. The really amazing fact is that with the proper vision and thus trick of using the second version, this abundance creates an abundance of effective sets without effective complements.

Rice's Theorem:

Let $[\Psi_1]$ and $[\Psi_2]$ be two disjoint, non empty and variant complete effective text sets.
Then they can not contain all texts, in fact neither all translates.
So we have a $|\Psi_0|$ that isn't in either set.

By the non emptiness let $T_1 \in [\Psi_1]$ and $T_2 \in [\Psi_2]$. Regard the:
$\{ BE ;\ B \text{ is translate} \wedge ((B \in [\Psi_1] \wedge E \in [T_2]) \vee (B \in [\Psi_2] \wedge E \in [T_1])) \}$ set.
This is effective so it is $[\Psi]$. Observe that for a B translate:
$B \in [\Psi_1] \rightarrow [\Psi] - B = [T_2]$ while $B \in [\Psi_2] \rightarrow [\Psi] - B = [T_1]$.
Now let $|\Psi_0|$ be the translate guaranteed by the Recursion Theorem for Ψ. Then:
$|\Psi_0| \in [\Psi_1] \rightarrow [\Psi_0] = [\Psi] - |\Psi_0| = [T_2] \rightarrow |\Psi_0| \in [\Psi_2]$ by variant completeness.
$|\Psi_0| \in [\Psi_2] \rightarrow [\Psi_0] = [\Psi] - |\Psi_0| = [T_1] \rightarrow |\Psi_0| \in [\Psi_1]$ by variant completeness.
So $|\Psi_0|$ can not be in either set if they are disjoint.

I want again to emphasize how absurd it is that Kleene found the Recursion Theorem but not this almost trivial consequence of it. It just goes to show that math is not about theorems but visions.
In text books, Rice's Theorem is always formulated for index sets and thus it seems as if this newer generalization would be the point of it. The real point starts with this consequence:

If $[\Psi]$ is variant complete, not empty, neither is the full set of texts then Ψ has no complement machine. Indeed, a complement would collect again a variant complete set and so these two would be the impossible two sets of the theorem above.
We might think that the variant completeness is a huge requirement that is hard to satisfy.
So now comes the second step of the real point that the variant completeness comes about by itself if we regard any program running property as Ψ.
Indeed, if we just care about how the programs run then all variants are collected.
The only still hidden problem is that we must get an effective Ψ and there is no rule that would tell if the such "property" is indeed effective. But the naïve concept of "watching a running" is enough to decide at once whether a property is effective or not.
To be truthful, here the best is to regard generations and not collections.
Let's see a few examples for number generations or lists:
Whether a program will list the number 100 is clearly effective. This was the example we already mentioned for the particular case we started with.
To list at least one prime or at least twenty primes is again effective.
We see them being fulfilled by watching the listed numbers.
But whether the number 10 will not be listed or no prime will be listed or no more than twenty is listed are not recognizable intuitively because these can not be assured from a beginning.
And indeed, these are the opposite sets of the previous ones and so their program sets are definitely not recognizable by Rice's Theorem.
Whether all primes are listed is again an intuitively not recognizable property because in a beginning we can see only finite many. But now the opposite, that is not listing all primes is also non recognizable intuitively because they all can still pop up after any beginning.
An even simpler case of this is whether a program will list exactly twenty primes.
Here a refutation comes about if we see already more than twenty primes but this still doesn't make either side observable. If it will list less, we can not be sure of this forever. And even if it does list exactly twenty, though we'll see this twenty, we can not be sure if one more comes.

Rice's Theorem can not be used for these intuitively dually non derivable program sets.
An interesting behavior is whether a list will have infinite or only finite many members.
This means the same as having arbitrary long texts or only up to a length.
For numbers this means having arbitrary large numbers or not.
The lists that have arbitrary long texts or arbitrary large numbers can be called unbounded.
Now we'll show that the programs for only these unbounded lists can not be listed.
A smart way to do this, is to prove that for any list of such unbounded lists there has to be some missing from our list. So we regard simply a list of lists. We must allow repetitions because different programs can produce same lists.

So let $\mathfrak{L}_1, \mathfrak{L}_2, \ldots$ be a list of unbounded lists.

The k-th text in $\mathfrak{L}_n$ could be denoted as $T_{n,k}$.

So the first text in $\mathfrak{L}_1$ is $T_{1,1}$.

Increase $T_{1,1}$ with a symbol to get a $T_{1,1}^{+}$

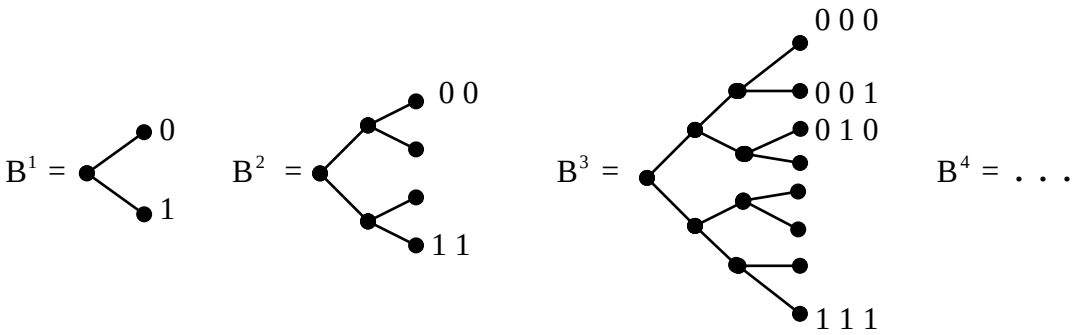
Now go in $\mathfrak{L}_2$ till the first T_{2,k_1} text comes that is longer than $T_{1,1}^{+}$.

Increase this again to get T_{2,k_1}^{+}.

Find the first text in L_3 longer than T_{2,k_1}^{+} and it is T_{3,k_2}.

And so on, we get an obviously increasing $\mathfrak{L}$ list that can not be identical with any $\mathfrak{L}_n$.

The first text of $\mathfrak{L}$ is already longer than the first in $\mathfrak{L}_1$ and so all texts in $\mathfrak{L}$ will be longer and so that first text is missing from $\mathfrak{L}$ definitely. But then the found longer text of $\mathfrak{L}_2$ will be missing again because we skip that length completely. And so on, all $\mathfrak{L}_n$ lists will have one definite member that can not be in $\mathfrak{L}$.



PASCAL Ø/

$\frac{25}{14} = 25 : 14 = 1.7\underbrace{8\,5\,7\,1\,4\,2}\,8\,5\ldots$ (repeating period)

```
 110
  120
    80
    100
     20
      60
       40
       120
```