Computational Biology and Bioinformatics
2025, Vol. 13, No. 2, pp. 72–83
https://doi.org/10.11648/j.cbb.20251302.14

SciencePG
Science Publishing Group

Research Article

Ways to Estimate the Minimal Specified Complexity of Reproduction

Waldean Schulz*

Independent Researcher, Spokane, United States

Abstract

Because of the difficulty of estimating the complexity of biological reproduction of even the simplest cell, the minimal complexities of several simpler self-replicating systems are computed. These include viruses (both biological and computer, which require additional complex resources to replicate). Also considered are a proposed self-reproducing factory and two conceptual self-replicators. Because even some of those are difficult to evaluate, very, very low bounds are computed. Still, in all cases, the minimal complexities are enormous-more than all the quantum state changes in the entire history of the whole known universe in all time. Therefore, because biological reproduction is more complex, all origin-of-life proposals, including RNA hypotheses toward the first self-reproducing cell, must demonstrate how at least such minimal complexity could have accumulated beyond using simply trial-and-error.

Keywords

Reproduction, Self-replication, Specified Complexity

1. Background

Reproduction is the defining feature of life. Another may be *metabolism*, but one could argue that an automobile has a form of the latter but not the former. Can we estimate a lower bound on the complexity of biological reproduction, which generates an offspring, which itself can further reproduce? Certainly, this question relates to the origin of biological life. The first lifeform needed to be self-replicating in addition to having self-sustaining metabolism. Self-replication (or reproduction) of any complex entity demands substantial coherent, specific, modular reproductive functionality in addition to any ancillary metabolic activity-whether the entity is biological or man-made.

Reproduction occurs in the biological world in various ways both sexually and asexually. Since estimating a lower bound on the complexity of biological reproduction is so difficult can we learn about the functions and complexity of simpler non-biological self-replication (presumably asexual)?

Although two biological viruses are briefly examined, this paper examines the minimal complexity of several examples of non-biological self-replication. It shows that for each example the complexity of information far exceeds the number of quantum state changes in the whole history of the entire universe. Because biological reproduction is at least as complex as the non-biological, there is the question of whether biological reproduction ever could have occurred naturalistically by trial-and-error.

Would you not like to own an automobile that replicates itself biannually? Presumably asexually! Self-replication of a

*Corresponding author: ds@deanschulz.com (Waldean Schulz)

Received: 7 November 2025; **Accepted:** 22 November 2025; **Published:** 11 December 2025

car would necessitate a whole built-in fabrication and assembly plant, including means to acquire and process all basic materials. How cumbersome that vehicle would be! What would its gasoline mileage or battery consumption be? Could you drive it on your next vacation or even just to the supermarket? Could it even be mobile? Could it maintain itself? In extreme contrast, your body is exceptionally mobile, acquires its basic materials, even repairs itself, is a product of reproduction, and is capable of further reproduction.

John von Neumann [1] conceived an *abstract* self-reproducing "automaton". However, so far, no engineer has constructed any kind of fully self-replicating *physical* kinematic device. For example, a "parent" 3-D printer can produce many parts for a "daughter" 3-D printer [2]. However, the "parent" cannot produce all the parts (such as electronic circuit boards), it cannot acquire the raw materials, and it cannot assemble the parts.

However, many very familiar, human-engineered self-replicating *software* entities do exist: namely, computer viruses, as well as their normally troublesome kin. What can they teach us about the minimum complexity of reproduction-biological or otherwise?

Because computing the specified complexity of the reproductive function of a cell would be exceptionally difficult, this paper examines several more primitive examples of reproduction, all simpler than cellular or multi-cellular reproduction. A minimum specified complexity will be computed for each example.

2. Computer Viruses

A *computer virus* (CV) is materially distinct from its inspiration, a *biological virus* (BV). Nevertheless, both viruses share some common *functional characteristics*: hence the shared name. Those basic characteristics are common among the gazillion variants of CVs and BVs. Much like biological viruses, computer viruses are characterized by specific sequences of software byte codes (instead of specific DNA/RNA sequences of codons or specific protein polymers of amino acids), which can reproduce and most often provoke a nefarious effect in an infected laptop. Both kinds of viruses are configured to replicate themselves, but both lack sufficient functionality to do so independently. Therefore, both kinds of virus must attach to (or embed in) an appropriate host, and both exploit the host's resources to accomplish self-replication. For a BV, the host is a live biological cell; for a CV, the host is a program to which the CV has attached and executing on a computer with an operating system. So, a lone BV is not alive in any usual sense of having self-sustaining metabolism; similarly, a CV is not a self-standing program and is not active until actuated by an executing host program on a computer.

This comparison of CVs to BVs is more than just an analogy. Both types of viruses exhibit specific, parallel, common, required functionality: identification of a target host to infect, attaching to/inside it, injecting into the host the code/codon sequence for a daughter virus, inducing the host to perform those instructions, and typically performing additional actions. Their respective hosts provide the "hardware" processor/ribosome that executes the instruction code/codon sequence to generate new copies of the virus and disseminate them.

So, a classic CV is not normally a self-standing executable computer program. Rather it must be attached to some normal program, its "host". When that host program is executed in the operating system of a computer, the CV's code is invoked; it locates one host program on the computer's hard drive; and it attaches a copy of itself (a "daughter" CV) to that other program. When that other program executes, the daughter CV is then invoked and does its mischief as did its parent CV, including generating a "granddaughter" CV. And so on.

Mark A. Ludwig, a physicist, was the author of books and a journal on computer viruses and artificial life. Although written three decades ago, his book *Computer Viruses, Artificial Life, and Evolution* [3] remains an instructive read. Ludwig wrote when the Worldwide Web, e-mail, and Internet file sharing were just reaching the tech-savvy public. His book was written in the early years of Microsoft Windows and the Apple Macintosh. At that time nearly all software was distributed on floppy diskettes, so diskettes were also the *medium* through which computer viruses propagated to personal computers.

Today most software is downloaded from the Internet, and most computer viruses spread through that medium. Further, now there are other, sophisticated variants called "malware" (malicious software), such as so-called worms, trojans, spyware, ransomware, and adware all inspired by CVs. The 1990's computer hardware/software technology about which Ludwig wrote (Intel 8086-powered PCs running the DOS operating system) seems "ancient" now. That suggests that his thoughts regarding computer viruses and artificial life might be passé. Nevertheless, I will show that his thoughts are instructive and lead to clear conclusions about the minimal complexity of self-replication / reproduction.

Ludwig [3] points out that CVs were never intended to be a simulation of primitive "artificial life" (AL). Nevertheless, they perhaps are unintentional, unbiased, primitive simulations of AL-specifically of reproduction; whereas explicit attempts to model reproduction, evolution, and the origin of life often reflect implicit bias. Explicit modeling may unconsciously and subtly include a bias toward what an evolution-oriented programmer wants to demonstrate [4]. So then, can CVs instruct us about the minimum complexity of biological reproduction, even in the minimal form as BVs and even when neither CVs nor BVs are fully self-reproductive on their own?

2.1. Shortest Known Computer Virus

Ludwig started a contest to design the smallest computer

virus to operate on the (then ubiquitous) Microsoft DOS operating system running on an Intel 8086 PC. The virus had to meet eight modest criteria. For example, the CV must be self-contained, and it must not modify the host program code to which it attaches itself. [3] The cash reward for the winner was $200 (back when that was more than play money).

The winning virus was written by a programmer with a pseudonym, who asked to remain anonymous. (I wonder whether it was Ludwig himself). The winning CV was called Companion-101. As the name suggests, it was only 101 bytes long (that is, 808 bits) and comprised 42 Intel 8086 microprocessor instructions, each 1 to 3 bytes long. 33].

Note that today no such short virus generally can infect software running on a modern operating system (OS): Windows, UNIX/Linux, MacOS, iOS, or Android OS. This is because now there are many hardware and software protections built in (file ownership, memory boundaries, checksums, …). Therefore, interactions with modern operating systems now require a CV to be a more sophisticated computer instruction sequence than a virus for a more primitive operating system like DOS. So, a contemporary CV must be a sequence of instructions much longer than 101 bytes. Yet, a modern OS does provide many more resources for the designers of viruses and other malware to exploit: resources like Wi-Fi access to the World Wide Web. In short, contemporary CVs must and will be more complex and larger than their DOS cousins.

Is there another potential 101-byte long (or shorter) code sequence which satisfies Ludwig's minimal criteria for a CV? Not likely: All the instructions are necessary, and their specific sequence is crucial for doing the job. It is an irreducibly complex, coherent entity. However, suppose that a clever programmer could swap two instructions in Companion-101 and not thwart successful duplication of itself. Or, the clever programmer might devise a different and even shorter CV, which self-replicates. Could a programmer exhaustively generate and test all possible code sequences of lengths less than 102 bytes to see if any qualify as a shorter CV? Note that the test would also require a host program to which to attach, and which would need to execute. As you will soon see, it is essentially impossible to use trial-and-error just to find a putative CV even among all possible sequences of just 50 bytes-only half as long as Companion-101. There simply are far, far too many potential byte-sequences to examine-even given all the time and resources our universe could provide.

Ludwig [3] explains why a shorter CV is unlikely to exist or, if so, why such would not likely proliferate. Nevertheless, for the sake of the argument, insanely and very generously suppose that there are 2^{100} possible bit sequences of less than 102 bytes that would satisfy Ludwig's CV criteria or even only some (but still replicate and proliferate). What fraction of all possible 101-byte and shorter byte sequences does that myriad of putative CVs present?

$2^{100}/256^{102} = 1 / 2^{716} = 2^{-716}$,

or approximately $1/10^{216}$, or 10^{-216},

So the putative 2^{100} shorter CVs are an exceedingly minuscule fraction of all possible code sequences less than 102 bytes.

However, what is the practical limit to the number of potential CVs we can test in practice? That is, to test whether an N-byte sequence replicates itself on a PC running DOS. How fast can we do each test? How much time is required? What is the shortest possible time to test all potential CVs?

A quantum state change is the shortest possible time period that makes sense in quantum physics: 10^{-45} second. Ridiculously suppose we could test whether each candidate byte sequence is a CV in just one quantum state change, which I call a Planck "tick". In reality, each test would require the equivalent of a plethora of quantum state changes. Let's estimate the number of all Planck ticks available in the history and volume of the universe.

2.2. The Physical Limit of the Total Possible Quantum Events

William Dembski [5] estimates a reasonable maximum on the count of discrete quantum state changes in our entire universe over its expected lifetime since the Big Bang through to its presumed eventual future demise: namely,

10^{150} events. How does he compute this? It is the product of 10^{80}, which is the count of estimated elementary particles in the universe; 10^{25} seconds, the age of the universe from Big Bang to its end; 1 / (10^{-45} second), the count of quantum state changes / second.

So, $10^{80} \times 10^{25} / 10^{-45} = 10^{150}$ quantum state changes.

Converting to base 2, we have $10^{150} = 2^{498}$ approximately. This number of all possible quantum events in the lifetime and space of our universe is also the same as the number of all unique linear sequences of 498 binary ones and zeros-far less than the unique sequences of 808 bits in Companion-101, the shortest known CV. That is surely a mind-boggler!

However, the number of discrete quantum state changes throughout the history and volume of the universe, 10^{150}, is exceedingly tiny in comparison to the number of all code sequences to be tested, namely 10^{216}, before bumping into a putative shorter CV-that is, just one of the 2^{100} very optimistically presumed CVs of length less than just 102 bytes. The number 10^{216} would be the number of quantum events during the existence of 10^{66} universes like ours. Are you convinced that these are well beyond unfathomable, mind-exploding numbers?

Alternatively, thinking in base two, 2^{498} is exceedingly tiny relative to 2^{716}. So, $1/2^{716}$ is the essentially impossible probability of finding one of the 2^{100} putative CVs among all possible software code sequences that are each less than just 102 bytes long. That is, testing all 101 byte (808 bit) and shorter code sequences to determine whether one of the 2^{100} putative CVs is truly a CV would require time resources far, far beyond what our whole universe could ever provide by many, many, many orders of magnitude.

The above also relates to Complex Specified Information (CSI) and the Explanatory Filter of William Dembski [5]. The set of all possible byte sequences less than 102 bytes long is clearly very *complex* in size. The *specification* is that the byte sequence must be a legitimate, self-replicating CV. So, every CV qualifies as CSI. Further, applying Dembski's Explanatory Filter, we find that the byte sequence of a CV neither occurs naturally, nor is it a product of chance. So, Companion-101 or any other CV must be *designed*. Hardly a surprise! Can any knowledgeable person think a computer virus occurred naturally and was not intelligently designed (albeit maliciously)? Surely not, considering all the above.

Because neither a CV nor a BV is fully capable to reproduce on its own but needs a host, each is only a portion of an entity which is independently capable to reproduce. So, the complexity of a fully capable, independently reproducing program will be much greater than the complexity of a simple CV. Therefore, all the above computations for complexity will be even huger for self-sufficient reproductive entities. Surely the same applies to lifeforms that can reproduce independently, like bacteria, being much more complex than CVs.

2.3. Can CVs Evolve

First, consider the likelihood, even within the old, simple DOS/PC environment, of a 101-byte CV evolving "from scratch". Assume that the digital memory of a flakey PC is very exceptionally erratic, so that mutation occurs frequently and randomly. Above I showed that the number of even shorter sequences of bytes grossly exceed the number of quantum events in the lifetime of our universe. So, it is essentially impossible for a 101-byte sequence to arrive randomly *de novo* and satisfy the self-replication specifications for a CV, because this is essentially equivalent to the CV testing / searching situation described above. The flakey PC is doing the search but is many orders of magnitude slower that our unrealistically fast testing rate of one test per quantum event. Further, the flaky PC is unlikely to preserve the CV it putatively could create!

Second, now consider the likelihood of a CV evolving from another functional non-CV program again on a very flakey PC with an extremely unstable memory and processor. Or maybe the PC is being bombarded with an extensive amount of nuclear radiation or is very near an MRI. We will assume the PC is in constant use for a very long indefinite period running many programs, including a putative "ancestor program" and any subsequent mutants of it. But no subsequent mutant is going to attach itself to (infect) another program *until it already has evolved all the basic self-replication functionality of a CV*. This would require that all generations of mutants keep subsequently adding to (or changing or rearranging) all the correct bytes in correct order of a proto-CV and then *protecting the accumulated correct bytes from further mutation*. This is equivalent to generating a CV *de novo*, as above, but building it already attached to an existing ancestor host program.

The ancestor program or its mutants likely will also already have instances of some or all the 8086 instructions of the CV-42 in the case of Companion-101, but very likely in jumbled order and separated. Perhaps, there would be 2 or 3 in the right order consecutively. But getting the correct 42 instructions is a very long shot. Even so, getting the correct order is so even more unlikely. There are over 10^{51} different orderings of the 42 specific instructions.

Each of the sequences of mutant programs would have to act as the host and so continue to be able to execute during the putative evolution. But the insertion of a detrimental mutation either in the host or in the attached nascent virus is so very likely-like a jump to a random location-effectively a stop code or continuous loop.

Ludwig [3] addresses the possibility of evolving Companion-101 from a shorter program, Stomper, which is a 66-byte subset of Companion-101. That is a difference of 280 bits. His conclusion is much like mine but with different reasoning.

For a larger CV, the above very, very slim potential of mutating from an existing non-CV program is even slimmer, because the larger combinatorics explode even more wildly. Of course, a larger CV more complex than Companion-101 can do "useful" actions, such as printing a nasty message to the user, gathering personal information, collecting marketing data, subjecting the user to ransom, or destroying data.

Second, the same reasoning holds for inserting or deleting one or more bytes or a combo thereof, because the combinatorics explode in our faces just as above. In all cases we are still considering a functional non-CV program, which must differ functionally from a CV's functionality and therefore differ by many bytes, because any useful program will differ substantially from a CV in the instruction byte sequence of each.

In seeming contradiction to the above, CVs do exist that *mutate*. But the mutations are very limited, specific, and explicitly engineered. The intent of these self-mutating CVs is to escape detection by anti-viral software, which detects a known CV by testing for a characteristic subsequence of bytes. (See a parallel here with biological antibodies?) If a CV by design incorporates extra "random junk bytes" or rearranges the coded instructions in the "daughter" CV, it can escape detection. But these mutations must be very intentionally and cleverly designed so they do not ruin the functionality (replication) of a daughter CV. So, the mutations assuredly exhibit *enhanced functionality,* added specification, and more code bytes. They certainly are not random mutations.

We are considering only software code sequences that simply replicate themselves (and that with the resources of a host); a code sequence that does that plus something more functionally useful would, of course, be far longer than 101 bytes-even more nearly impossible to evolve on a flakey, error-prone (mutating) personal computer.

3. Intelligence Short-cuts the Vast Set of All Possible Programs

So, how did Companion-101 arise? From the above we can clearly conclude that it was not a result of a random or natural process. Certainly, there is no tendency for the computer's memory to naturally form a CV. Its creator did not (could not!) use trial and error. That does not exclude the creator from first producing and testing a few dozen attempts that did not self-replicate properly before creating a fully functional CV. Alternatively, maybe the creator successfully generated a 131-byte CV, but from that got *intelligent insight* in how to shorten the CV to 101 bytes. In any case, intelligence produced Companion-101 (as well as the bevy of CVs that have plagued MS-DOS programs in the past or the CVs which threaten modern computer operating systems now). The same holds for any instance of software: it requires goal-directed, creative engineering.

So, intelligence-guided creation of CVs (and all purposeful software) very impressively short-cuts the enormous combinatorics that dwarf even the number of possible quantum events in the lifetime of the universe. Teleological, goal-driven intelligent agency very quickly culls the plethora of all potential byte sequences by many, many, many orders of magnitude. How an engineer does that is a marvel of how the mind works. How long did it take to create Companion-101? A day? A week? Even if it took a whole year (surely not!), working steadily 8 hours per day, that is a mere 10^7 seconds compared to discovering it by trial and error.

Of course, any significantly useful program (think about a computer operating system) is orders of magnitude longer and more coherent than Companion-101: typically, many megabytes in size.

4. Can CVs and BVs Be Fruitfully Compared

Is it reasonable to compare the combinatorics, probabilities, features, functionality, criteria, and evolution of CVs and BVs? A lazy skeptic can easily dismiss any comparison by simply pointing out the very basic physical difference. A person could argue that a CV is not physical-but is informational-because the byte-sequence can be represented equivalently in so many differing media: optical, magnetic, solid-state memory, ink on paper, symbolically in assembly language, as binary digits, and so on. In contrast, a BV comprises physical molecular proteins and DNA/RNA. Nevertheless, as will be presented later, Fritas and Merkle (2004) and Mignea (2012, 184) present common generalized design requirements that list the minimally required modular features, functionality, and coherent interdependencies thereof for a physical self-reproducing entity.

5. SARS-CoV-2 Virus Spike Protein

A CV plus the computer executing its bytes as instructions is surely far less complex than the simplest BV plus the host cell translating the BV's nucleotides into strings of amino acids for daughter BVs. For instance, a SARS-CoV-2 BV causing COVID-19 has 29 types of proteins, which cumulatively comprise nearly 10000 amino acids (not counting the repeated instances of the proteins such as the multiple spike proteins on the BV's surface). Each instance of one variant of its spike proteins alone has 672 amino acids (a.a.). Assuming only 4 bits per a.a., that is 2688 bits, not counting the information in the makeup of its amino acids and assuming 4+ bits are equivalent to specifying one of 20 amino acids. [6] Assume some redundancy (either of two or more different amino acids that may occupy some locus in the protein and still provide the same functionality). So, assuming on average that only 1.5 bits are needed to specify each amino acid of the 672, the Shannon information in the spike protein alone is still over 1000 bits. So, just the spike protein is at least as complex as the whole of Companion-101. So, whether you like it or not, the spike protein or some very similar variant was intelligently designed.

6. A Minimal Biological Virus (BV)

Bacteriophage Phi-X174 [7] has a genome size of only 5386 nucleotides. Counting 2 bits per nucleotide, it has 10772 bits of information. 10772 bits by itself exceeds 498 bits by far, even if only half the virus provides the RNA/DNA for replication. So, it is questionable whether it could have evolved *de novo*. Yet, it is not self-replicating on its own; it requires a whole bacterium and its RNA/DNA replication machinery.

7. Von Neumann's Self-replicating Abstract Machine

In a series of lectures in 1948 and 1949 and in subsequent papers, John von Neumann [1] proposed a cellular automaton which reproduced itself. Many variants and implementations thereof followed. One implementation of von Neumann's self-replicating automaton was by Umberto Pesavento [8], which will be studied here as an example. It is clearly even more complex and involved that the CV cited above. One reason is that the rules for transitioning of the 32 cell states are simpler that the machine instructions of the 8086 computer involved with the simplest CV, Companion-101, so more "machinery" was required to be implemented in the cellular transition rules. Just to illustrate the complexity, note the following 2-D diagram of the cellular automaton in Pesavento's implementation: Figure 1.

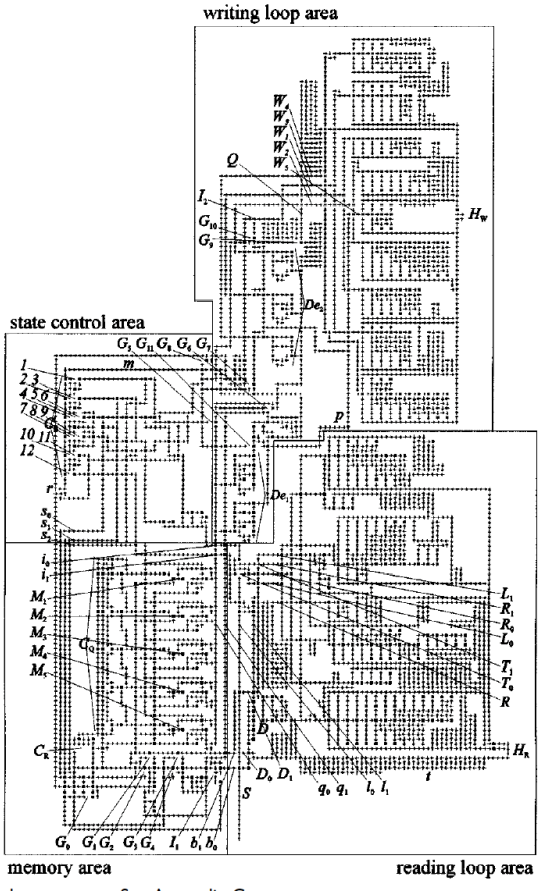

Figure 1. *An Implementation of Von Neumann's Self-Replicating Abstract Machine.*

This clearly includes more than 6500 cells, where each cell contains $\log_2(29)$ bits of information, or > 29000 bits. So too this example of self-replication implies that reproduction/replication cannot be generated by evolutionary trial-and-error but rather by intelligent design.

8. NASA's Self-reproducing Lunar Factory

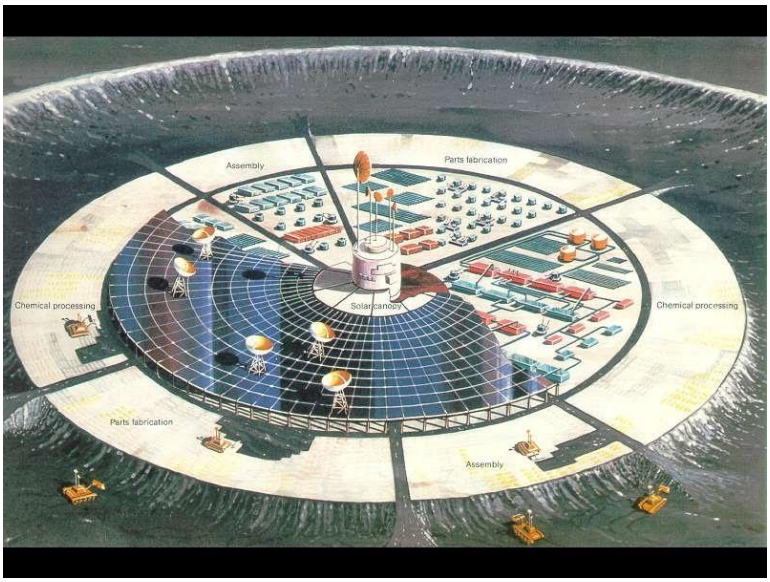

Figure 2. *NASA's Self-Reproducing Lunar Factory.*

NASA sponsored conceptual design proposals for a lunar self-replicating factory. One, described by Robert A. Fritas [9], required enormous amounts of materials (63 to 145 tons) and power (0.5 to 12 megawatts) to achieve all necessary functionality. See the illustration below. It is certainly not mobile. Given the impractical enormity of the self-replicating lunar factory proposal, Fritas [10] has subsequently written about general kinematic self-replication, but he now focuses on self-replication in nanotechnology. In any case, Fritas has documented the numerous required modules, their functionality, and their interrelationships for any form of mechanical or molecular self-replication. See Figure 2.

The project provided an accompanying knowledge graph, shown below in Figure 3. I will not defend its correctness or completeness but will assume it more or less captures the steps and effort to build the Lunar Self-Replicating Factory.

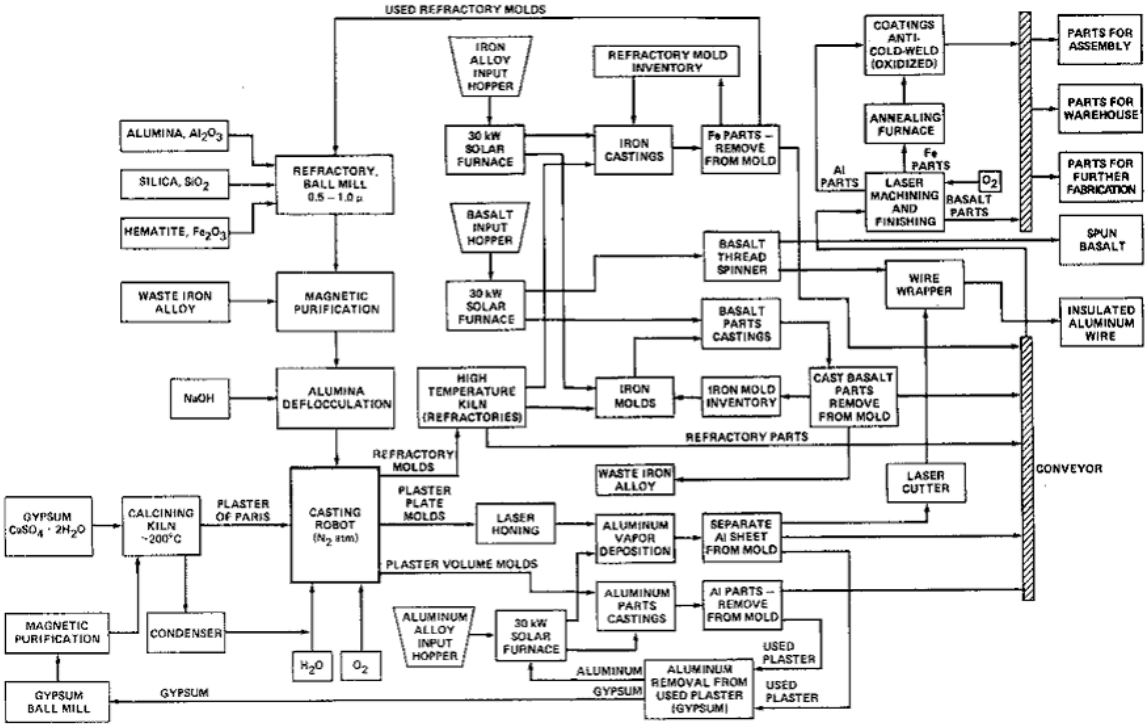

Figure 3. *The Knowledge Graph of NASA's Self-Reproducing Lunar Factory.*

A list of the nodes corresponding to the above is shown below. Because we have no idea what the information content is for each node, we will simply count the bits in the characters of the label in each node. This surely is orders of magnitude less than what would be the actual information content of each node. So, the computations appended to the nodes are a very low lower-bound on the actual information content.

Table 1. *Computation of the Minimal Specified Complexity of NASA's Self-Reproducing Lunar Factory.*

NASA Lunar Self-Replicating Factory			
nodes	**node info bits**	**out links count**	**node info * #out links**
alumina	56	1	56
silica	48	1	48
hematite	64	1	64
waste iron alloy	128	1	128
NaOH	32	1	32
gypsum	48	1	48

NASA Lunar Self-Replicating Factory			
nodes	**node info bits**	**out links count**	**node info * #out links**
calcining kiln	112	2	224
gypsum magnetic purification	224	1	224
gypsum ball mill	128	1	128
condenser	72	1	72
refractory ball mill	152	1	152
iron magnetic purification	208	1	208
alumina deflocculation	168	1	168
casting robot	104	3	312
water: H_2O	80	1	80
oxygen: O_2	80	1	80
iron alloy input hopper	184	1	184
30 kW iron solar furnace	192	2	384
basalt input hopper	152	1	152
30 kW basalt solar furnace	208	2	416
high temperature kiln	168	3	504
laser honing	96	1	96
aluminum input hopper	168	1	168
aluminum solar furnace	176	2	352
refractory mold inventory	200	1	200
iron castings	104	1	104
iron molds	80	1	80
aluminum vapor deposition	200	1	200
aluminum parts casting	176	1	176
aluminum removal from used plaster	272	2	544
iron parts - remove from mold	232	3	696
basalt thread spinner	168	2	336
basalt parts castings	168	1	168
iron mold inventory	152	1	152
separate Al sheet from mold	216	3	648
aluminum parts - remove from mold	264	2	528
cast basalt parts - remove from mold	288	3	864
coatings anti-cold-weld	184	1	184
anneaing furnace	128	1	128
laser machining & finishing	216	2	432
wire wrapper	96	1	96
laser cutter	96	1	96
conveyor 1	80	1	80

NASA Lunar Self-Replicating Factory			
nodes	**node info bits**	**out links count**	**node info * #out links**
conveyor 2	80	1	80
parts for assembly	144	0	144
parts for warehouse	152	0	152
parts for further fabrication	232	0	232
spun basalt	88	1	88
insulated Al wire	136	0	136
totals	7200	63	10824
units	bits	count	bits

Note that 10824 bits far exceeds 498 bits (or 150 decimal digits), the number of quantum state changes in the entire history of our universe. So self-replication involving these steps seems to be indisputably complex, and of course it is specified, so it is complex specified information (CSI).

Because of the shear difference in scale (size, power, time) between the lunar factory and biological organisms and the materials, there is the question of whether this example has any relevance to asexual biological reproduction. In fact, it may be even more relevant than software self-replication, because it requires temporal assembly of *physical* resources and does not assume an underlying host machine that is itself not reproduced.

To avoid the potential difference-in-scale difference-in-material counterarguments, now consider a more generic approach to self-replication, which can be considered independent of scale and material.

9. Mignea's Simplest Self-replicator (SSR)

To study a different model of self-replication, consider the generic simplest self-replicator (SSR) specified by Arminius Mignea [11]. It considers all the minimum required functionality of any physical self-replicator. He presents a knowledge graph (Figure 4), although he does not specify directional links. Assume that half the links are outbound and half are inbound. As above, we will assume that the content of each node is simply the information represented by its label. Of course, that is likely many orders of magnitude less than a detailed description of the particular actions of each node. So, once again we are dealing with a very low lower-bound for the actual information within self-replication. The knowledge graph information content evaluation is as in Table 2.

Table 2. *Computation of Minimal Complexity of the SSR.*

SSR Knowledge graph			
node names	**node info bits***	**count of links ****	**node info x #links / 2**
Bill of Materials	136	4	272
Materials & Parts ident.	192	3	288
Construction Plan	136	7	476
Construction Status	152	5	380
Input Flow	80	4	160
Supply Chain	96	8	384
Scaffolding Growth	144	6	432
Transport	72	6	216

SSR Knowledge graph			
node names	**node info bits***	**count of links ****	**node info x #links / 2**
Fabrication	88	11	484
Materials Extraction	160	7	560
Enclosure Growth	128	6	384
Manipulation	96	8	384
Energy Generation	136	4	272
Fabrication Contlro & Assembly	208	6	624
Output Flow	88	3	132
Recycling	72	5	180
Cloning	56	4	112
Division	64	4	128
Replication	88	2	88
Communication and Notification	240	38	
totals	2432	71	5956
units	bits	count **	bits

* We ignore any further information in the nodes themselves: just the bits in the title.

And we ignore any information in the various relations (links).

** Links are counted twice: two per node; so divide by 2, if links are one-way.

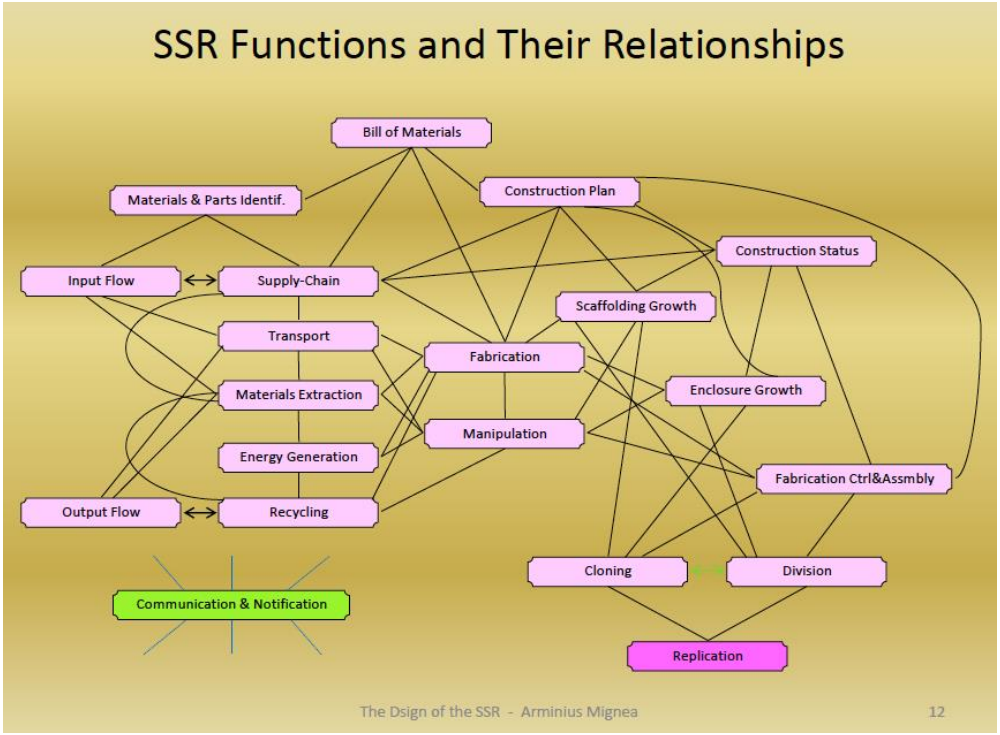

Figure 4. *The Knowledge Graph of the SSR.*

Again, we have a very low lower-bound, 5956 bits, which far exceeds 498 bits. So, this is a very low lower bound on any self-reproducer.

10. RNA Replicase

Consider the case of RNA replicators, which are hypothesized to be in the prebiotic chain leading to a protocell. Experiments which studied the mutations occurring after hundreds of replications of RNA replicase evolution *begin* with a long RNA replicase derived from an existing virus.[12] This RNA enzyme and all functional mutants are at least 200 nucleotides long. This suggests that there is a minimum length threshold for general replication to function-just like for computer viruses.

Mutants occur regularly because RNA is characteristically unstable, especially in water. Assume there are 1 billion variants of an RNA replicase. Even then, the complexity of RNA replicases is still huge, considering they comprise at least 200 nucleotides:

$$4^{200} / 10^{9} = 2.6 \times 10^{111}. \quad (1)$$

or, $\log_2(2.6 \times 10^{111}) = 192$ bits of information.

While this complexity (1) is not as large as the complexities computed above and is less than 10^{150}, it still is huge enough to raise the question of how such information could be accumulated during the prebiotic development of life on Earth, during;

5×10^5 years $\times$ 31556736 sec/year $\times$ 10^{45} quantum state changes/sec = 1.6×10^{58} quantum state changes.

Clearly $10^{111} >> 10^{58}$. So, the likelihood of forming such a long *initial* replicator enzyme is essentially zero within the prebiotic period on Earth. However, for any sort of cumulative prebiotic evolution to occur, complex RNA replicators, their mutants, or any other polypeptide replicators are mandatory from the outset.

11. Conclusion

The case studies examined-computer viruses, two biological viruses, von Neumann's automaton, NASA's lunar factory, Mignea's SSR, and RNA replicase-demonstrate the consistent principle that reproduction is impossible without specified, modular complexity usually at a level far beyond the probabilistic resources of our universe.

1) Computer viruses show that even minimal replication requires precise coding and host exploitation. A single corrupted instruction would prevent propagation [13].
2) Biological viruses, including one of the simplest, exhibit complexity beyond Dembski's very generous probability threshold.
3) Von Neumann's automaton illustrates that reproduction requires organization: a constructor, duplicator, and controller, working in coordinated interdependence [1].
4) NASA's lunar factory proposal underscores that large-scale physical reproduction requires carefully integrated systems of acquisition, fabrication, and self-maintenance, all of which must be pre-designed [14].
5) Mignea's SSR highlights that generic self-replication only occurs within environments explicitly engineered with rules and reproductive functions already in place [15].
6) RNA Replicase, which can replicate RNA polypeptides, does have a lower complexity by itself than the examples above, but it requires availability of sufficient correct nucleotides in appropriate concentrations in an appropriate environment, which are of unknown additional complexities themselves. Whether RNA replicase would operate in the wild like it does in carefully controlled and isolated lab situations is not obvious.

Across these diverse examples, the same conclusion emerges: self-replication is not the byproduct of random trial-and-error but the outcome of foresight, planning, and highly modular specification. The information content required for reproduction generally exceeds 10^{150}, the maximum number of quantum events possible in the history of the universe. Consequently, the probability that biological reproduction could have arisen spontaneously through unguided processes is effectively zero.

Therefore, the origin of life requires explanation in terms of purposeful, intelligent agency. The comparisons show that intelligence not only vastly surpasses a mindless trial-and-error evolutionary search in generating replication, but that replication without intelligence has never been observed, simulated, or engineered.

Abbreviations

SSR Simplest Self-Replicator

Author Contributions

Waldean Schulz is the sole author. The author read and approved the final manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] Von Neumann, J. Burks, A. W. *Theory of Self-Reproducing Automata*; Univ. of Illinois Press, Urbana, 1966.

[2] Fox News, This New 3-D Printer Can Print Its Own Parts; May 23, 2016; https://www.foxnews.com/tech/this-new-3d-printer-can-print-its-own-parts

[3] Ludwig, M. A. *Computer Viruses, Artificial Life; and Evolution*; American Eagle Publishers, Tucson, AZ, 1993.

[4] Marks, R. J., Dembski, W. A., Ewert, W. *Introduction to Evolutionary Informatics*ˆ, chapter 6, p 243; World Scientific Publishing, New Jersey, 2017.

[5] Dembski, W. Design Inference, Cambridge University Press, Cambridge, UK, 1998, p 209.

[6] Almehdi, A. M., *et al.* SARS-CoV-2 spike protein: pathogenesis, vaccines, and potential therapies, 2021 Oct; 49(5): 855-876. https://doi.org/10.1007/s15010-021-01677-8 Epub 2021 Aug 2.

[7] Wikipedia, Phi X 174, accessed 2025/8/15 https://en.wikipedia.org/wiki/Phi_X_174

[8] Pesavento, U. An Implementation of von Neumann's Self-Reproducing Machine, Princeton University. Princeton, NJ, 2000.

[9] Freitas, R. A., Gilbreath, W. P. (Eds.). Advanced automation for space missions. NASA Conference Publication 2255. NASA Scientific and Technical Information Branch, 1980.

[10] Freitas, R. A. Jr, Merkle, R. C. Kinematic Self-Replicating Machines; Landes Bioscience, Georgetown, TX, 2004; http://www.MolecularAssembler.com/KSRM.htm

[11] Mignea; A. Developing Insights into the Design of the Simplest Self-Replicator and its Complexity: Part 2-Evaluating the Complexity of a Concrete Implementation of an Artificial SSR; Engineering and the Ultimate, ed. Jonathan Bartlett, et al, Proceedings of the 2021 Conference on Engineering and Metaphysics, Blyth Institute Press, Broken Arrow, OK.

[12] Mizuuchi, R., Furubayashi, T. & Ichihashi, N. Evolutionary transition from a single RNA replicator to a multiple replicator network. Nat Commun 13, 1460 (2022). https://doi.org/10.1038/s41467-022-29113-x

[13] Cohen, F. Computer viruses: Theory and experiments. Computers & Security, 6(1), 1987, 22-35.

[14] Freitas, R. A. Jr, Zachary, W. B. A Self-Replicating, Growing Lunar Factory; Proceedings of the Fifth Princeton/AIAA Conference, May 18-21, 1981, Jerry Grey and Lawrence A. Hamdan (Eds.), p 24; http://www.rfreitas.com/Astro/GrowingLunarFactory1981.html

[15] Rasmussen, S., Bedau, M. A., Chen, L., Deamer, D., Krakauer, D. C., Packard, N. H., Stadler, P. F. *Transitions from nonliving to living matter*. Science, 303(5660) 2001, 963-965.

Biography

Waldean Schulz is a retired software and systems engineer. He completed his PhD in computer science in 1976 at the University of Colorado. His past employers include Intel, Language Resources, Ball Aerospace, PixSys / Image Guided Technologies, Nervonix, and Retül / Specialized Bicycles. He has implemented programming language compilers, 3-D real-time LED trackers, medical devices, and bicycle fitting systems. Schulz also is a patent agent registered with the US Patent Office. He holds 15 patents. He is author of several refereed technical and biological papers including An Engineering Perspective on the Bacterial Flagellum: Part 1, 2, and 3.

Research Field

Waldean Schulz: engineering approach to biology, how to compute specified complexity, 3-D tracking of surgical instruments, imaging subdural nerves by impedance sensors, bicycle fitting.