3 Chapter 3: Beyond Exact Self-Replication

So far in the first two chapters, we have only considered the scenario where a machine is to build an exact copy of itself, ideally in a reliable, error correcting manner. This setting is crucial for the original motivating thought exercise concerning the colonization of Mars. In such a scenario, it is extremely important that the lineage of machines preserves the exact same functionality: both to ensure efficient progress in the colonization process, and to maintain safety guarantees. In particular, we must rule out the possibility that a potentially malfunctioning offspring develops behavior that is adversary to the terraformation effort, or even worse, to humankind. Following the terminology from [22], we refer to such self-replicators as the *maker-replicators*.

3.1 Von Neumann's Vision of Open-Ended Evolution

However, von Neumann envisioned a far more ambitious goal than exact self-replication: the definition of a system in which artificial "organisms" evolve and increase in complexity in a potentially unbounded manner, in analogy with living systems. We refer to such a process as *open-ended evolution*. In such a scenario, it is essential that an organism possesses the capacity of creating an offspring of greater complexity than itself. Crucially, von Neumann's replicator design already outlines a logical pathway toward such a capability. A key feature of the von Neumann self-replicator is its construction universality: it is not limited to constructing copies of itself, but can build any machine whose description is provided on its tape, regardless of that machine's complexity (subject to feasibility within the medium). If the system containing his universal self-replicator was subject to noise and small perturbations, one can imagine that the tape undergoing mutations, with some of these changes leading to descriptions of machines more complex than the parent. Importantly, such changes in the tape description are manifested phenotypically in the offspring. One may therefore say that von Neumann's design supports *inheritable mutations*, thereby providing a natural pathway toward increasing structural complexity. Von Neumann articulated this vision vividly in his Illinois lectures [25, Part I, 5th lecture, p. 78]:

"Anybody who looks at living organisms knows perfectly well that they can produce other organisms like themselves. This is their normal function, they wouldn't exist if they didn't do this, and it's plausible that this is the reason why they abound in the world. In other words, living organisms are very complicated aggregations of elementary parts, and by any reasonable theory of probability or thermodynamics highly improbable. That they should occur in the world at all is a miracle of the first magnitude; the only thing which removes, or mitigates, this miracle is that they reproduce themselves. Therefore, if by any peculiar accident there should ever be one of them, from there on the rules of probability do not apply, and there will be many of them, at least if the milieu is reasonable. But a reasonable milieu is already a thermodynamically much less improbable thing. So, the operations of probability somehow leave a loophole at this point, and it is by the process of self-reproduction that they are pierced.

Furthermore, it's equally evident that what goes on is actually one degree better than self-reproduction, for organisms appear to have got more elaborate in the course of time. Today's organisms are phylogenetically descended from others which were vastly simpler than they are, so much simpler, in fact, that it's inconceivable how any kind of description of the later, complex organism could have existed in the earlier one. It's not easy to imagine in what sense a gene, which is probably a low order affair, can contain a description of the human being which will come from it. But in this case you can say that since the gene has its effect only within another human organism, it probably need not contain a complete description of what is to happen, but only a few cues for a few alternatives. However, this is not so in phylogenetic evolution. That starts

from simple entities, surrounded by an unliving amorphous milieu, and produces something more complicated. Evidently, these organisms have the ability to produce something more complicated than themselves."

We summarize this crucial property of the von Neumann self-replicator in the following informal definition, and note that a detailed discussion of the significance of von Neumann's construction in the context of artificial life can be found in [15].

Definition 3.1 (Informal definition of self-replication with evolutionary potential)**.** *A self-replicator has evolutionary potential if it is capable of producing an offspring of higher complexity than itself (for a so far unspecified notion of complexity), and if the offspring in turn also has this property, thereby potentially giving rise to a lineage of replicators of increasing, unbounded complexity. This capability may be realized through suitable external perturbations of the replicator's tape content.*

The definition above involves several deliberately vague notions, most notably that of a replicator's complexity. As emphasized by von Neumann himself, until this term is properly defined, "none of this can get out of the realm of vague statement" [25, p. 80].

Non-Trivial Self-Replication The need to define the complexity of self-replicators becomes apparent for one more reason: that is, to formally justify the power of von Neumann's construction and distinguish it from trivial instances of self-replication. Consider the following cellular automaton analogy of the crystal growth, a canonical example of a self-replicating process commonly regarded as trivial. This two-dimensional CA has two states $\{0, 1\}$ and the following simple local update: a single cell in the state 1, surrounded by the quiescent state 0, spreads at each time-step in all directions, as illustrated in Figure 1. Clearly, although the single cell produces copies of itself, this process does not elucidate any of the underlying mechanisms of self-replication. Intuitively, it stands in stark contrast to von Neumann's construction. Formally distinguishing trivial cases of self-replication from non-trivial ones, however, remains a challenging open problem.

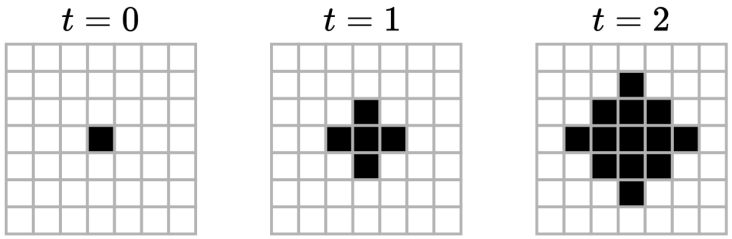

Figure 1: A canonical example of trivial self-replication in a cellular automaton.

In his editorial remarks, Burks suggests to resolve this by emphasizing that the VNSR is Turing universal. One should agree that Turing universality is a sufficient condition for a certain level of complexity, however a series of later works showed that Turing universality is a property independent of non-trivial self-replication. In one direction, we will show in this chapter a construction of a compact, non-trivial self-replicator called the Langton loop, which is not computationally universal. In a later chapter, we will discuss the converse: the existence of Turing universal systems that are unable to facilitate non-trivial self-replication.

To summarize, we are left with a series of properties of self-replicating systems:

$$\text{non-trivial}, \text{computation universal}, \text{construction universal}, \text{has evolutionary potential}$$

In the following chapters, we will discuss results that rigorously define what it means for a dynamical system to be computationally universal, and relate this notion to non-trivial self-replication. Defining formally the remaining notions introduced above—including that of a self-replicator's complexity—largely remain open problems: their formal characterization, as well as the establishment of precise relationships between them, is still unresolved. For now, we highlight one important relationship that has already emerged implicitly in the discussion. Orginally, von Neumann did not introduce construction universality for evolutionary purposes; indeed, it served primarily as a mechanism to ensure exact self-replication. However, as we have discussed, it remains the only known sufficient condition for evolutionary potential.

We conclude this discussion by citing von Neumann one last time: "One cannot define the concept of complication correctly until one has seen in greater detail some critical examples, that is, some of the constructs which exhibit the critical and paradoxical properties of complication. There is nothing new about this. It was exactly the same with conservation and non-conservation properties in physics, with the concepts of energy and entropy, and with other critical concepts. The simplest mechanical and thermodynamic systems had to be discussed for a long time before the correct concepts of energy and entropy could be abstracted from them."

Fortunately, von Neumann's seminal work on self-replication inspired a rich line of future self-replicator constructions, spanning a wide range of sizes and capabilities across various dynamical systems, most notably cellular automata. The rest of this chapter is dedicated to giving an overview of such examples. As the notions introduced above have yet to be formalized, examining concrete constructions will hopefully equip the reader with valuable intuition regarding their interplay.

3.2 Second Wave of CA Self-Replicators: Universal Constructors

Von Neumann's original construction leads to a CA pattern of an extensive size, estimated to span between 50,000 and 200,000 cells (according to [21, p. 241]). At the time of its conception, such a construction was far beyond the capabilities of available computers, making direct implementation effectively impossible. This motivated a second wave of research aimed at simplifying the self-replicator's design while preserving its construction universality. Simplifications could proceed along several, often competing, dimensions. One may reduce the number of CA states or the neighbourhood size, typically at the cost of increasing the spatial extent of the replicator. Alternatively, one may aim to reduce the overall size of the replicator, at the expense of introducing additional CA states or increasing the length of the replicator's description. A central practical goal of this line of work was to reduce the replicator's size sufficiently to allow explicit computational implementation.

From an implementation perspective, the realization of a CA self-replicator can be understood as consisting of three conceptually distinct steps, listed in Table 1 in order of increasing difficulty.

Implementation	Criterion
Spatial	A CA configuration is constructed that spatially realises the full replicator structure, possibly with a short instruction tape sufficient to demonstrate construction of a simple object.
Informational	The full replicator is implemented together with an instruction tape encoding its complete self-description, regardless of whether the full self-replication cycle is practically feasible.
Temporal	A complete self-replication cycle is explicitly simulated, from initiation to the production of a functioning replicant.

Table 1: Progressive criteria for CA self-replicator implementations.

A temporal implementation is not merely a visually appealing demonstration; it also plays a crucial role in verifying the correctness of a construction. Given the high complexity of universal constructors, purely theoretical construction outlines—even when sound in their core ideas—often overlook subtle errors, as becomes apparent upon careful examination of the literature, and as noted, for instance, by Hutton [11].

In Table 2, we provide a non-exhaustive list of some of the most significant contributions.

Author	**Contribution**
Codd [8] (1968)	Introduced a universal self-replicator in an 8-state cellular automaton with von Neumann neighbourhood and rotational symmetry. The construction was too large for a spatial implementation at the time.
Devore [10] (1970s)	Designed a replicator using a modification of Codd's rule, spanning 19,105 non-quiescent cells. A temporal implementation is available in Golly, requiring around 1.02×10^{11} time-steps for replication.
Pesavento, Nobili [17] (1995)	Implemented von Neumann's original self-replicator using a 32-state CA. A later version appears to replicate successfully in Golly, with a full cycle taking approximately 6.34×10^{10} time steps.
Hutton [11] (2010)	Identified and corrected issues in Codd's construction and provided an informational implementation. Estimated that observing a full replication cycle in Golly would take on the order of 1000 years.
Buckley [4, 3] (2007–2013)	Provided the first informational implementation in von Neumann's original 29-state CA. No full replication time estimate is given; Golly simulation crashes after $\sim 1.2 \times 10^{10}$ time steps.

Table 2: Major implementations of universal constructor–type CA self-replicators.

Many additional intriguing constructions can be found in Golly's database [24] or on online forums such as `https://conwaylife.com`. However, these constructions have generally not been disseminated through a peer-reviewed process, which makes their precise capabilities difficult to verify. We nevertheless encourage the reader to explore their implementations in Golly.

To summarize, all the works of the second wave preserve the replicator's construction universality, as well as a clear separation between the machine and the tape. It follows in a straightforward way (though we rely on rather intuitive, informal arguments) that all such replicators do have the evolutionary potential. A price to pay for this powerful property is the replicator's size: Despite decades of refinement, among published works, the constructions of Devore and of Nobili and Pesavento appear to yield temporal implementations with some of the shortest known self-replicating cycles, which nevertheless remain on the order of 10^{11}.

Codd's Construction For our purposes, namely, understanding the structural ideas that enable efficient self-replication, it is unnecessary to analyse all the engineering innovations of the works mentioned above. Instead, we focus on a particular design principle introduced by Codd. This principle was later adopted in several constructions and became a foundational ingredient of the *third wave of self-replicator designs*, whose aim is to obtain extremely compact replicators with short replication cycles.

Codd retained the von Neumann neighbourhood but imposed an additional constraint on the local rule, known as *rotational symmetry*, which further simplifies the cellular automaton architecture by reducing the number of distinct neighbourhood cases that must be handled explicitly.

Informally, the local rule assigns the same output to a neighbourhood configuration and to all of its rotations (by 90°, 180°, and 270°). We formalize this notion below.

Definition 3.2 (Symmetry of a CA local rule)**.** *Let $\mathcal{A} = (S^{\mathbb{Z}^d}, F)$ be a d-dimensional cellular automaton with neighbourhood $\mathcal{N} \subset \mathbb{Z}^d$ and local rule $f\colon S^{\mathcal{N}} \to S$. A map $c\colon \mathcal{N} \to S$ is called a* neighbourhood configuration *of $\mathcal{A}$, and the set of all such configurations is denoted $S^{\mathcal{N}}$.*

Let G be a group acting on $\mathcal{N}$, representing a set of geometric symmetries of the neighbourhood. This action induces an equivalence relation on $S^{\mathcal{N}}$: for $c_1, c_2 \in S^{\mathcal{N}}$, we write $c_1 \sim_G c_2$ if there exists $g \in G$ such that $c_1 = c_2 \circ g$. The automaton $\mathcal{A}$ is invariant under G *if*

$$c_1 \sim_G c_2 \implies f(c_1) = f(c_2),$$

that is, if the local rule depends only on the symmetry class of the neighbourhood configuration.

Codd's automaton is invariant under the cyclic group of order four generated by rotation by 90°. This symmetry substantially reduces the number of distinct neighbourhood configurations for which the local rule must be specified, as the following exercise shows.

Exercise 3.3 (Unique neighbourhoods in Codd's CA)**.** *Consider a two-dimensional cellular automaton with a von Neumann neighbourhood and 8 states.*

(1) In the absence of any symmetries, how many outputs of the local rule must be specified?

(2) If the CA is invariant under rotations by 90°, 180°, and 270°, how does this number change?

We now give a more detailed description of Codd's construction [8], or, more precisely, of Hutton's corrected version thereof [11]. To reduce the number of states, Codd eliminated the directional transmission states used by von Neumann; Codd's eight states are shown in Figure 2(a). Instead of four oriented transmission states, Codd introduced a single *channel* state responsible for signal propagation: a single row of cells in the channel state functions as a bidirectional transmission line. The direction of a signal is determined by a trailing state 0; for example, the adjacent states 07 represent a value 7 moving to the right, while 70 represents the same value moving to the left. As in von Neumann's design, specific instructions—such as extending the channel by one cell—are implemented as specific sequences of trailing signals.

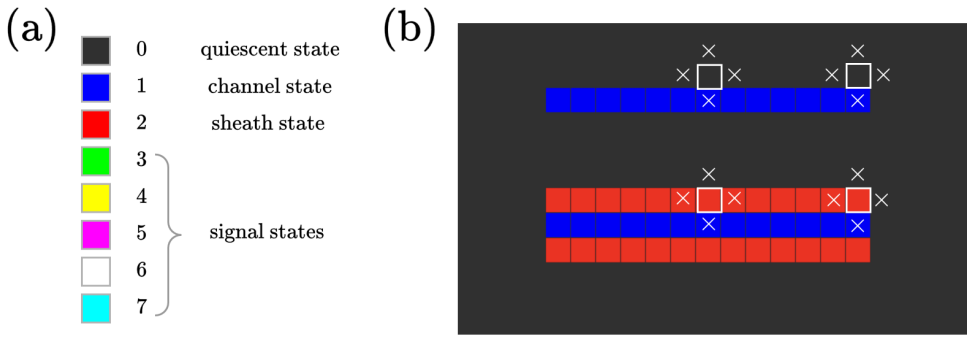

Figure 2: (a) Overview of 8-states of Codd's CA. (b) In the top configuration, the edge cell above an unsheathed channel cannot distinguish whether it is at the channel's end, or in the middle of it. In contrast, in the bottom sheathed channel, the cell is able to distinguish such situations.

An interesting element in Codd's construction is the sheathing: a line of cells in the "sheath" state, that cover every transmission channel from both sides. The purpose of the sheathing is

nicely described in Hutton's revision [11] that we paraphrase. Suppose we have a horizontal row representing the bidirectional channel, and there is a signal arriving from left to right, instructing the channel to make a left turn, and extend by one cell. In the unsheathed case, the cell responsible to grow has no awareness whether it is at the end of the channel, or in the middle of it. However, in the sheathed case, it has enough information to distinguish these two cases. We illustrate this in Figure 2 (b).

We note that, even in the sheathed case, the designated cell in Figure 2(b) does not have sufficient information to determine whether it lies at the end of a channel or at the corner of a channel turning to the right. In this situation, the cell can further deduce its correct position and its role in extending the channel in the desired direction by observing the interaction of the instruction stream with the channel's end: the trailing signal either propagates past the cell (the corner case) or terminates (the end case). This behaviour is illustrated in Figure 3. Codd's solution of adding a sheath to every channel is thus an elegant way to increase the "spatial awareness" of cells while keeping the automaton's neighbourhood size and number of states small.

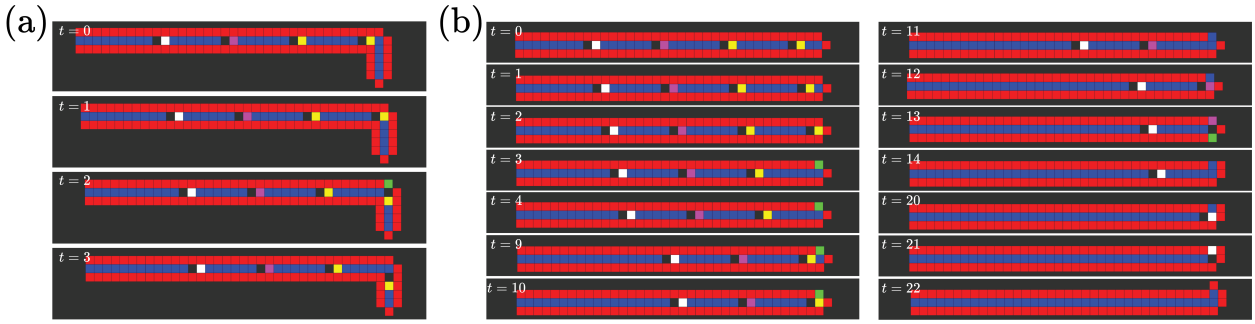

Figure 3: A signal 40-40-50-60 is sent through the sheathed channel, with instructions to extend the channel by one cell, making a left turn. (a) The cell in the corner gets temporarily activated at $t = 2$ by the signal, learning by the trailing 0 passing through that it is not at the end of the channel, and getting deactivated again. (b) In this case, the top cell above the channel learns at $t = 3$ that it is indeed at the end of the channel, and stays activated until further signals arrive, with the growth completed at $t = 22$.

A particular building block that will be useful in later constructions is the so-called periodic emitter, illustrated in Figure 4.

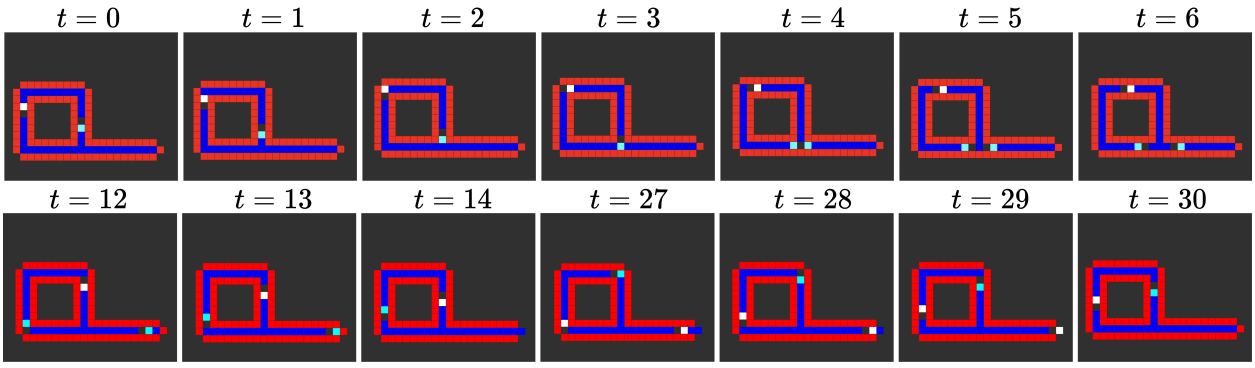

Figure 4: The signal 70-60 instructs the channel to grow one cell forward. Top: the signal loops around and duplicates each time it passes the junction. Bottom: one cycle of the channel's growth.

It is based on a sheathed channel that forms a loop with a single path extending from the loop. Any signal contained in the loop will be indefinitely circling around, and every time it reaches the junction, the signal will get duplicated to travel down the path. If we combine the signal 70 followed by 60 with appropriate spacing, this combination will result in an instruction to extend the path by one cell forward.

Similarly, different signal combinations can instruct the path to be extended so that it makes a left, or a right turn. We suggest the reader to pause and already think of ways this design could be used to construct a relatively simple self-replicating loop. The solution is presented in the next section.

3.3 Third Wave: Non-Trivial Self-Replicators

In 1984, when none of the universal constructors had yet been implemented, not even spatially, Christopher Langton set out to design a self-replicator of significantly smaller size, one whose replication cycle could be fully implemented and observed. To achieve this, it appeared necessary to sacrifice some of the functionality present in the second-wave replicators. In particular, Langton argued that although construction universality is a sufficient condition for genuine, non-trivial self-replication, it is not a necessary one.

The self-replicator he designed is not construction universal, nor is it computation universal. As such, he had to again face the tricky question: what criterion should be used for non-trivial self-replication that would clearly separate his construction from trivial cases such as the afore-mentioned crystal growth? To this, he suggested the following informal condition: if the self-replicator contains its description and uses this information twice during the self-replication process– once for translation, and once for transcription– it should be considered a non-trivial self-replicator. This criterion has been adopted by a rich line of subsequent works as a sufficient condition for non-trivial self-replication. It remains unclear whether this separation between an organism and its description, and the description's double usage is the only necessary organization that can achieve genuine non-trivial self-replication. We note that an interesting result in this direction that suggests the necessity of such organization was developed in [14].

The self-replicator Langton designed is neither construction universal nor computationally universal. As a result, he was again confronted with an important question: what criterion should be used to define non-trivial self-replication in a way that clearly distinguishes it from trivial cases such as the afore-mentioned crystal growth? In response, he proposed the following informal condition: if a self-replicator contains a description of itself and uses this information twice during the replication process—once for translation and once for transcription—then it should be regarded as a non-trivial self-replicator. This criterion has since been adopted as a sufficient condition for non-trivial self-replication. It remains unclear, however, whether this separation between an organism and its description, together with the description's double usage, constitutes the only organizational principle capable of achieving genuinely non-trivial self-replication. We note that an interesting result suggesting the necessity of such an organization was developed in [14].

Langton succeeded at constructing a CA with 8 states, and within it, a self-replicating loop spanning 86 non-quiescent states, and a replication cycle of length 151. His loop exhibits precisely the double usage of its description, and can create exact copies of itself in a remarkably efficient time. However, the loop does not have any functionality beyond the exact-self replication. This inspired a so-called *third wave of replicator designs*, which aimed at preserving the compact size of CA self-replicators, while increasing the range of function the replicator is capable of; culminating in designs that can perform arbitrary computations (i.e., they are computation universal but not construction universal).

We now present several concrete examples of compact self-replicator designs, serving as representative case studies of the principles outlined above. We begin with a non-exhaustive overview of some of the most significant contributions in Table 3, and subsequently discuss several of them in greater detail. For a more complete overview, we refer the reader to Sipper's survey [21] and to the more recent one by Sayama and Nehaniv [20].

Author	**Contribution**
Langton [13] (1984)	Constructs a self-replicating loop in a CA with 8 states, von Neumann neighbourhood, and rotational symmetry. The loop spans 86 non-quiescent cells and has a replication cycle of length 151. It is neither construction universal nor computationally universal.
Byl [5] (1989)	Further reduces the loop size by designing a CA with 7 states, von Neumann neighbourhood, and rotational symmetry. The resulting loop spans 12 cells and has a replication cycle of length 25. It is is neither construction universal nor computationally universal.
Tempesti [23] (1995)	Designs a loop capable of constructing a pattern inside the looped channel. A demonstrated example writes the letters `LSL` and is embedded in a CA with 10 states and Moore neighbourhood. The loop spans 148 cells and has a replication cycle of length 304. It is is neither construction universal nor computationally universal.
Perrier, Sipper, Zahnd [16] (1996)	Equip the loop with an additional tape storing an arbitrary program executed after self-replication. The CA has 9 states and a von Neumann neighbourhood; the loop spans 158 cells and has a replication cycle of length 235. It is computationally universal but not construction universal.
Chou, Reggia [7] (1998)	Show how self-replication can be used for computation by designing loops that solve instances of the SAT problem. Partial solutions are passed to offspring loops, while loops encoding incorrect candidates are eliminated. The loop is neither construction universal nor computationally universal.

Table 3: Selected examples of third-wave cellular automaton self-replicators.

Langton loop [13] Similarly to Codd, Langton works within a two-dimensional cellular automaton with a von Neumann neighbourhood, eight states, and rotational symmetry. Indeed, Langton's loop can be viewed as an adaptation of Codd's periodic emitter. Langton starts from a looped sheathed channel with a single path extending from it, exactly as in Codd's periodic emitter, and aims to design a sequence of signals circulating within the loop that encodes the following instructions. The individual steps are illustrated in Figure 5.

(a) Extend the construction arm by a sufficient length, then turn left.

(b) Iterate the previous step until the construction arm folds onto itself and forms a new loop.

(c) Copy the signal sequence into the newly formed loop.

(d) Dissolve the connection between the parent loop and the offspring loop.

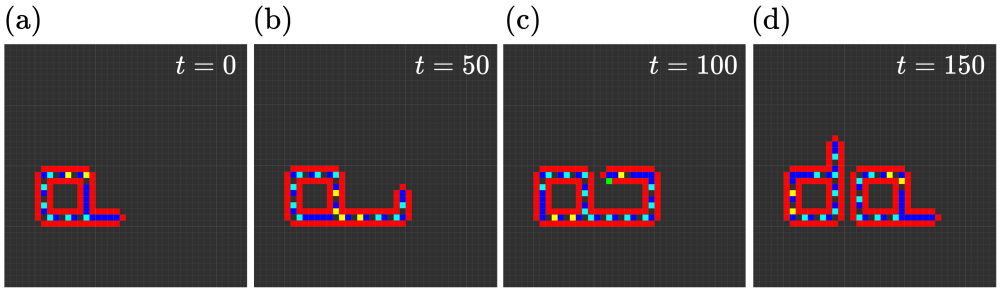

Figure 5: Self-replication stages of Langton's loop.

The main challenge lies in designing a signal sequence that encodes all of the above instructions while still fitting entirely within the loop itself. To address this, Langton observes that since the signal sequence is duplicated every time it reaches the junction, and since the loop has four identical sides, it suffices to encode instructions for constructing a single side of the loop. The combination of rotational symmetry and signal reuse thus allows a single compact instruction sequence to generate the entire structure. These instructions are then naturally iterated as the signal circulates around the parent loop and splits at each junction.

In essence, the remaining task is to modify the local rule of Codd's cellular automaton so that the instruction sequences encoding the growth of the loop's single side, fit within the loop. For a loop with a side length n, this sequence consists of n instructions to extend the construction arm forward, and one instruction to make a left turn. This is quite straightforward: Langton preserves the states of Codd's CA and amends the local rule so that the signal to prolong a channel one step forward has length two (encoded by states 70), and a signal to turn left has length five (encoded by states 40140).

Tempesti loop An interesting elaboration of the loop-based design is due to Tempesti [23], who defines a loop capable of self-replication and, in addition, of constructing a pattern specified by a program contained within the loop.

Tempesti considers a two-dimensional cellular automaton with Moore neighbourhood and, in its simplest form, five states. The loop's self-replication follows a principle similar to Langton's, but uses only a single inner sheath layer. In contrast to Langton's design, the loop's description is compressed to an extreme: the design relies on a CA rule in which the sheath grows largely autonomously, without explicit instructions from the parent loop. The only instruction transmitted from the parent is a single signal of length one, indicating when the growing sheath should make a left turn. This extreme compactness leaves most of the circulating program available for specifying functionality beyond self-replication. As demonstrated by the author, by introducing three additional states and extending the local rule accordingly, the program can construct a fixed pattern, representing the letters LSL, inside the loop around which it circulates (Figure 6).

Although this design is clearly capable of construction beyond self-replication, it is important to note that each enrichment of functionality requires the introduction of new CA states and a corresponding modification of the local rule. As the author explicitly remarks, it therefore remains unclear whether construction universality can be achieved for a fixed CA rule. Similarly, it is unclear that this loop possesses evolutionary potential. Modifications of the circulating program may manifest phenotypically in offsprings that construct different internal patterns, but such changes do not affect the size or morphology of the loop itself. Since the design does not allow a loop to construct a larger loop, the length of the program, and hence the range of constructible patterns,

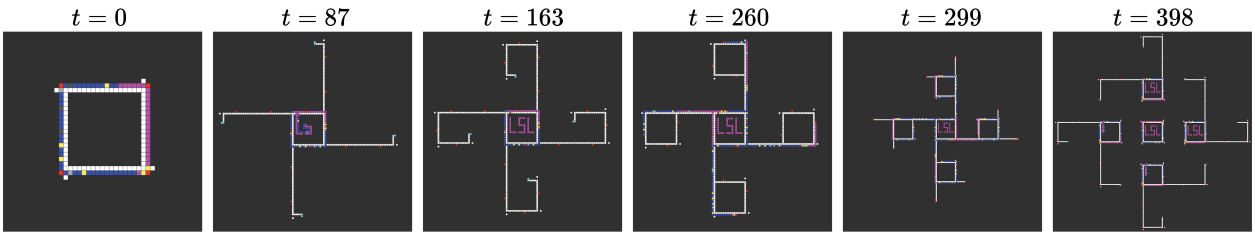

Figure 6: Tempesti loop. $t = 0$: The offspring's shape is encoded just by the four red states signifying left turns. Rest of program encodes the construction of LSL pattern inside the loop. $t = 87$: Red states travel down the construction arm, LSL pattern is being built simultaneously. $t = 260$: Transcription. $t = 299$: Offsprings self-replicate. $t = 398$: Offspring construct the pattern.

remains bounded.

To conclude, for self-replication to support evolutionary potential, it must allow for morphological diversity: offsprings must have the capacity to grow larger in order to accommodate genomes of increasing complexity. While this constraint is less apparent in biological organisms—whose physical size far exceeds that of their genome—it becomes particularly restrictive in loop-based designs, where the organism is approximately as large as its genome.

Perrier-Sipper-Zahnd loop The authors claim that construction universality of a self-replicator is the main cause of the replicator's prohibitive size. They succeed at constructing a self-replicating loop that is capable of universal computation but not universal construction.

Their design consists of three parts: the loop (based on the design of Codd and Langton), the program, and the data. Both the program and data a represented as tapes, attached to the bottom left and bottom right corners of the loop respectively. Whereas the data tape can grow in size with no upper bound, the program stays immutable. Due to this design, replication happens only in one direction, each loop creating an offspring to its right. Each loop first self-replicates, creating an exact copy with the exact same program and tape attached, and subsequently, it carries out the computation defined by its program.

The self-replication is done is three stages, first the actual loop is built, analogously to a Langton's loop, afterwards, the program tape is copied, and finally the data tape as well.

The computation is based on a compact universal system, akin to a Turing machine introduced by Wang. The sheath layer is used to accommodate a program head, which highlights the current active state, as well as the data head. It is also used to transmit signals between the two heads, represented as sequences of pulses. We illustrate the process of self-replication and computation in Figure 7.

Chou and Reggia: Computation through Self-Replication Whereas in the previous example, the exact content of the tape is copied to the offspring, Chou and Reggia design loops that pass on to their offsprings modified tapes with partial solutions to a computational problem. In some sense, they implement recursion via self-replication. Loops with solutions that do not seem promising are killed, essentially terminating certain branches by artificial selection.

Concretely, the authors consider the satisfiability problem. The initial loop, again replicating on principles similar to the Langton's loop, carries in its program loop not only the instructions to self-replicate but also encode a particular Boolean formula in the conjunctive normal form. At each self-replication step, two offsprings are produced, one of them replacing the parent, and a

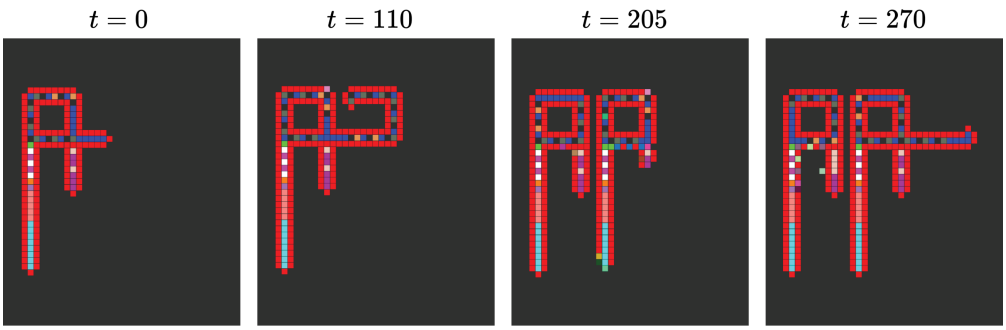

Figure 7: The process of self-replication and computation of the Perrier-Sipper-Zahnd loop. At $t = 0$ the original loop, at $t = 110$ the construction of the offspring loop, $t = 205$ program and data tape are being copied, $t = 270$ parent loop is computing with the program head sending a sequence of signals to the data head through the sheath channel; offspring is self-replicating. Images generated by Golly [24].

specific free variable assigned the value 0 in one offspring, and 1 in the other. Throughout the CA grid are spread the so called "monitors", each representing a specific clause. These monitors, when encountering a loop, check whether the partial solution the loop represents satisfies that clause. If this is not the case, the loop is erased and none of its offsprings are produced. In this way, the authors show examples where the system finds the correct solutions.

Discussion We highlight a crucial observation: although these compact designs are impressive and introduce a wide range of new capabilities, none of them reintroduces construction universality, which appears to this date to be the only known mechanism that supports open-ended evolutionary potential in artificial self-replicators. Indeed, the replicator designs we have discussed focus primarily on exact self-replication, and it remains unclear whether they can support inheritable mutations that manifest as phenotypical changes in the offspring, potentially giving rise to a replicator lineage of unbounded complexity. This limitation marks a sharp departure from von Neumann's original vision, and stands in tension with the role evolutionary potential plays in biological systems. To emphasize this point, we paraphrase NASA's working definition of life, which places central importance on the ability of a living system to evolve:

Life is a self-propagating system capable of undergoing adaptive evolution.

Let us highlight one more important difference between the second and thirds wave of CA replicators. Von Neumann's self-replicator design does not rely on any particular property of the cellular automaton local rule that is explicitly "hardcoded" to enable self-replication. Instead, von Neumann chose his elementary components in a deliberately general manner, ensuring primarily that they were expressive enough to support a sufficiently rich collection of higher-level functional building blocks. It is difficult to formalize this distinction precisely, but one might refer to it as *von Neumann's universal self-replication recipe*.

In contrast, there appears to be a general tendency that, as replicator constructions become more compact, the cellular automaton local rule becomes increasingly fine-tuned to enable such compact self-replication. This observation suggests that compactness may be achieved either by reducing functionality or by embedding replication mechanisms directly into the underlying dynamics. Along this axis, one can view self-replicating systems as forming a spectrum, with von Neumann's construction at one extreme and cellular-automaton crystal growth—where replication

is entirely hardcoded into the underlying dynamics and does not elucidate any mechanisms of self-replication—at the other. This perspective raises an intriguing question: what is the true reason that second-wave self-replicators were so large? Is it their rich functionality, namely construction universality, or rather the fact that they deliberately avoid exploiting any special features of the cellular automaton local rule?

3.4 Evolving Self-Replicators

Eventually, the aim to design self-replicators that can indeed produce more complex offsprings resurfaced, and resulted in a line of interesting works which typically design a system, together with an initial generation of self-replicators (or just a single one), as well as mechanisms by which these organisms can interact, in order to observe some emergent phenomena; that is, phenomena that are not explicitly encoded into the physics of the system. We briefly present three iconic systems designed to study evolving self-replicators.

Evoloop Sayama is one of the first authors who reintroduces a certain capacity for inheritable mutations into the CA self-replication loop design. His loops, again based on Langton's design, have two interesting properties:

1. When construction arms of two loops collide, they can influence each other and modify the signal sequence they transmit, potentially resulting in a modified offspring.
2. When a loop's construction arm collides with another loop, or vice versa, a special "dissolving state" propagates through both loops and erases them.

As such, this design provides an analogy to inheritable mutations and natural selection. Interestingly, when Sayama initializes the CA configuration with a single evoloop, the resulting dynamics leads to a diverse species of such loops, as shown in Figure 8.

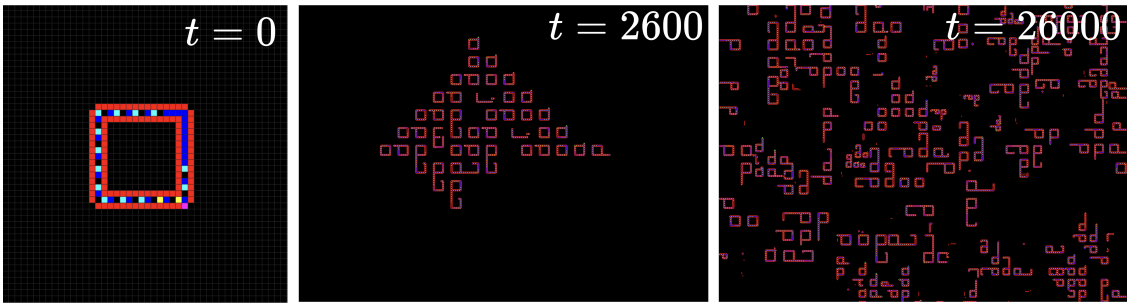

Figure 8: Evoloops. Initial loop ($t = 0$) that replicates ($t = 2600$) and diversifies ($t = 26000$).

However, after observing the evolution of such a system, progressively smaller loops come to dominate the space. This makes it very unclear whether the loops possess genuine evolutionary potential.

Sayama's work was later elaborated by Salzberg [19] who aimed precisely at mitigating the degeneration of the loop sizes. He introduced new hostile environments into the dynamics, such as the presence of pathogens that dissolve loops, or external mechanisms for loop removal. He demonstrates that these mechanisms can lead to an increase in loop size, but this growth appears to be only temporary before the system stabilizes.

Below, we briefly introduce two famous models of evolving self-replicators that are not embedded directly in a cellular automaton. Rather, the replicators take the form of self-referencing programs executed by an external CPU, akin to the quines we discussed in Chapter **??**.

Tierra Tierra is a virtual computational world designed by the theoretical biologist Thomas S. Ray in 1991 [18], with the explicit goal of creating an artificial world exhibiting open-ended evolution. The system is constructed so that programs can self-replicate, undergo mutations, and evolve new behaviours through "natural selection".

The world operates on a custom low-level programming language called Tierran, consisting of 32 primitive instructions. Each digital organism is a contiguous sequence of Tierran instructions stored in a shared block of virtual RAM (the "soup"). Every organism is associated with its own virtual CPU containing registers, a stack, and an instruction pointer. At each step, the virtual CPU reads the instruction at its current position in memory, determines what operation it represents, performs that operation, and then moves to the next instruction. After each instruction, the instruction pointer is automatically incremented, unless it is modified by a particular instruction (e.g., jumps). Importantly, organisms may read and execute instructions anywhere in memory, but they may write only within their own allocated memory block. This helps to protect the integrity of each organism.

Although organisms are executed sequentially on a single physical processor, Tierra implements time slicing: each organism receives a limited number of instruction executions per scheduling cycle. This aims to approximate parallel dynamics. Replication is not explicitly represented by a single low-level instruction. Instead, organisms must implement a self-referencing code using primitive instructions. Ray hand-designed an initial self-replicating program consisting of 80 instructions. Mutations are introduced in two ways: (i) stochastic errors during execution, and (ii) random bit flips applied to memory. Further, Tierra includes a "reaper" mechanism. When memory usage exceeds a threshold, organisms are removed according to a queue-based system influenced by their age and tendency to produce errors. Importantly, when an organism dies, its memory is deallocated but its code remains in the soup, where it may still be read or executed by other organisms.

As Ray shows, Tierra leads to interesting dynamics, producing a diverse soup of organisms, that evolve through inherited mutations, and develop, for instance, into parasites: programs that execute a part of their host's code, or hyper parasites: organisms that drive parasites to extinction; see Figure 9. Some organisms grow in size, in some experiments, reaching more than 23,000 instructions. However, later critiques show that each run eventually reaches a stasis where novelty ceases to grow.

Avida Avida is a digital evolution platform developed in the mid-1990s by Chris Adami, Charles Ofria, and colleagues [1] as a successor to Tierra. Like Tierra, Avida consists of self-replicating computer programs that are executed in a shared virtual environment under a fixed instruction set. Each digital organism contains its own code for replication and competes for computational resources. However, Avida introduces an explicit fitness structure: organisms are rewarded for performing specific computational tasks (such as logical operations). Successfully performing these tasks grants additional CPU time, thereby increasing reproductive success. In this way, Avida couples evolutionary dynamics to measurable functional outputs rather than relying solely on ecological interactions. This makes Avida particularly suitable for controlled experiments on adaptation and the evolution of complex functions, but it also means that evolutionary innovation is shaped, and ultimately also bounded, by externally defined rewards rather than emerging purely from organism–organism dynamics.

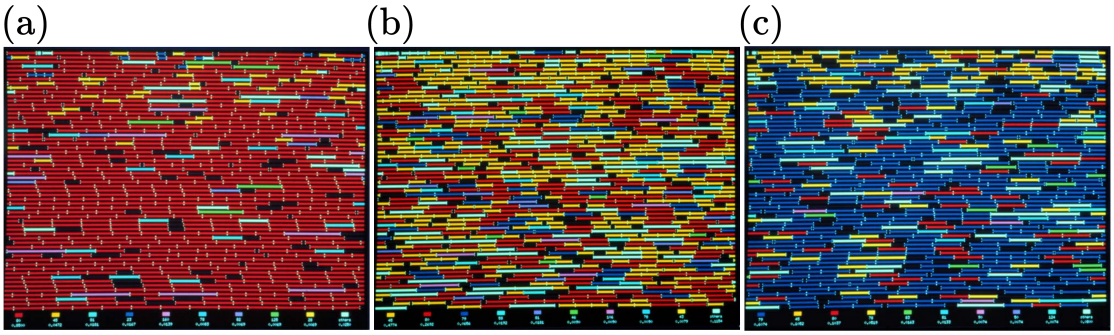

Figure 9: Evolution of a Tierra population, images from Marc Cygnus [9]. (a) Hosts (red) dominate with a few rare parasites (yellow). (b) Parasites dominate, with a few immune hosts (blue) appearing. (c) Immune hosts now dominate memory.

3.5 Emergence of Self-Replicators

In all the works we have discussed so far, self-replicators have been carefully hand-designed, and the initial configuration has been seeded with them. Such a setting already brings a plethora of open problems, including the identification of necessary and sufficient conditions for a self-replicator to have evolutionary potential, and the search for a design that realizes this potential. An equally fascinating question is how self-replicators might emerge from simpler co-evolving components that were not carefully hand-crafted. We are far from understanding this question theoretically; nevertheless, we briefly mention a line of work that aims to explore this direction.

Chou and Reggia's growing loops In [6], Chou and Reggia design loops that can produce larger offsprings. Moreover, the authors demonstrate that such self-replicating loops can emerge from randomly generated initial configurations. Below, we give a high-level overview of their design.

Chou and Reggia carefully craft a CA rule, which allows the existence of very small self-replicating loops. Their self-replication principle is again similar to that of Langton's loop. Here, however, the authors minimize the size of the loops by completely removing the sheathing, and encoding both the signal to advance a loop's construction arm forward or to turn it left by a single cell. As a result, the shortest self-replicator consists of a 2×2 pattern of non-quiescent cells. We note that the sheath removal is possible since the authors consider a larger, Moore neighbourhood.

Simplistically, if a $n \times n$ loop interacts with a cell in a designated growth state, the authors arrange the CA local rule in such a clever way that the loop's resulting offspring can be of size $(n+1) \times (n+1)$. Therefore, if a loop is surrounded by a rich enough environment, its offsprings can grow in size, potentially indefinitely. Another clever mechanism the authors implement in the CA local rule is the detection of failed self-replication, which can result in the annihilation of the loop with potential debris left behind, therefore leaving free space for more successful replicators.

Now, how can such replicators emerge from randomly generated initial configurations? The authors implement this through a clever trick. Essentially, the CA state space can be written as $S \times \{0,1\}$ where S is a finite set and the $\{0,1\}$ represents the "bound bit". We will call every state of the form $(s,1)$ an active state. Essentially, if a configuration is initiated with only active states, the CA rule is designed to mimic the dynamics of Conway's Game of Life:

- If a quiescent cell has exactly three active neighbours, it will itself become active in the next epoch. Its active value will be determined quasi-randomly based on the state of its neighbours.

- If an active cell has exactly two or three active neighbours, it will stay active; otherwise, an active cell will return to the quiescent state in the next epoch.

Once the smallest 2×2 replicator forms, since it is so small that each cell is aware of all the loop's components, all cells in this 2×2 region change their active state to 0 at once. Once this happens, the CA rule designed for self-replication is followed by this pattern and as a result, the loop starts replicating.

To summarize, the authors designed a CA rule with the following strategy for the emergence and growth of self-replicators:

1. Phase 0: Initialization: A configuration is initialized randomly with only active states.
2. Phase 1: Quasi-random exploration: The CA dynamics mimic Game of Life, and explore a rich set of local patterns, until the minimal 2×2 replicator is generated.
3. Phase 2: Self-Replication Transition: Since this loop is so small that each of its cells is aware of the whole pattern, all cells constituting the loop switch all at once into the inactive state. Cells in the inactive state follow the part of the CA rule table designed for self-replication.
4. Phase 3: Growth and Diversity: The loop replicates and interacts with the environment, which leads to its potential growth and the coexistence of self-replicating loops of varying sizes.

This work exhibits intriguing dynamics, yet both the emergence of self-replicators and their growth is enabled by encoding very clever mechanisms into the rule set. As such, it is not obvious whether the loops could ever develop any functionality beyond replication, or whether they could develop different morphologies.

Emergence of Self-Replicating Programs The emergence of quines was also studied in [12] and [2]. All these works share the same crucial property: the laws of physics (in this case, the programming language and ways programs are executed) designed by the authors allow the existence of extremely short replicators. Due to their small size, the replicators have a non-trivial probability of emerging during the initial phase, which mimics random exploration. After such a short replicator is "randomly sampled" through the dynamics, it starts to take over the system and is easily detected, since we can find many copies of it within the system. In this light, an important question arises: did the first biological self-replicators emerge as "random guesses" from the underlying dynamics or did they emerge as a composition of some proto-self-replicating parts? To the best of our knowledge, it remains an open question whether any artificial system models the latter scenario.

Remark 3.4. *The range of artificial examples of the emergence and growth of self-replicators in various media is vast, and we have by no means presented an exhaustive list of such examples, as that is clearly beyond the scope of the present notes.*

3.6 Chapter Summary

The examples presented in this chapter form a diverse and, at times, difficult-to-compare landscape. This is not accidental: even the authors of these constructions often lack clear, shared formal criteria by which their self-replicators can be evaluated. As a result, we have relied on a range of partially informal notions—such as non-triviality, computational universality, construction universality, and evolutionary potential, to describe and contrast these systems. The diversity of examples makes it

evident that there is a pressing need for more precise definitions that clarify what these constructions do, and equally importantly, what they do not demonstrate.

A natural guiding principle emerges from this observation: before attempting to understand the "biology" of artificial worlds populated by self-replicators, one must first understand their underlying "physics," namely the constraints and possibilities imposed by the dynamical system itself. As we have seen in this chapter, cellular automata provide a sufficiently expressive notion of physics to realize a wide variety of logically distinct self-replication mechanisms. This justifies our focus on cellular automata, and related dynamical systems, in the second part of these notes. There, we turn from individual constructions to the rules themselves, with the goal of identifying which dynamical systems are complex and expressive enough to support interesting phenomena. Specifically, we will study how this expressiveness relates to their dynamical behaviour and computational capacity.

References

[1] Adami, C., Ofria, C., and Collier, T. C. Evolution of biological complexity. *Proceedings of the National Academy of Sciences 97*, 9 (2000), 4463–4468.

[2] Alakuijala, J., Evans, J., Laurie, B., Mordvintsev, A., Niklasson, E., Randazzo, E., Versari, L., et al. Computational life: How well-formed, self-replicating programs emerge from simple interaction. *arXiv preprint arXiv:2406.19108* (2024).

[3] Buckley, W. A Catalog of Self-replicating Cellular Automatons, 04 2013.

[4] Buckley, W. R. On the complete specification of a von neumann 29-state self-replicating cellular automaton. Manuscript, 2007. Unpublished manuscript, originally written 1996–2004.

[5] Byl, J. Self-reproduction in small cellular automata. *Physica D: Nonlinear Phenomena 34*, 1 (1989), 295–299.

[6] Chou, H.-H., and Reggia, J. A. Emergence of self-replicating structures in a cellular automata space. *Physica D: Nonlinear Phenomena 110*, 3 (1997), 252–276.

[7] Chou, H.-H., and Reggia, J. A. Problem solving during artificial selection of self-replicating loops. *Physica D: Nonlinear Phenomena 115*, 3 (1998), 293–312.

[8] Codd, E. F. *Cellular Automata*. Academic Press, New York, 1968.

[9] Cygnus, M. Almond overview images. `https://tomray.me/pubs/images/`, n.d. Images reproduced with permission; credit required to Marc Cygnus.

[10] Devore, J., and Hightower, R. The devore variation of the codd self-replicating computer. Presented at the Third Workshop on Artificial Life (Artificial Life III), Santa Fe, New Mexico; unpublished manuscript, 1992.

[11] Hutton, T. J. Codd's self-replicating computer. *Artificial Life 16*, 2 (04 2010), 99–117.

[12] Koza, J. R. Spontaneous emergence of self-replicating and evolutionarily self-improving computer programs. *Artificial life III 17* (1994), 225–262.

[13] Langton, C. G. Self-reproduction in cellular automata. *Physica D: Nonlinear Phenomena 10*, 1 (1984), 135–144.

[14] Marletto, C., and Deutsch, D. Constructor theory of life. *Journal of the Royal Society Interface 12*, 104 (2015), 20141226.

[15] McMullin, B. John von Neumann and the Evolutionary Growth of Complexity: Looking Backward, Looking Forward. *Artificial life 6* (02 2000), 347–61.

[16] Perrier, J.-Y., Sipper, M., and Zahnd, J. Toward a viable, self-reproducing universal computer. *Physica D: Nonlinear Phenomena 97*, 4 (1996), 335–352.

[17] Pesavento, U. An implementation of von neumann's self-reproducing machine. *Artificial Life 2*, 4 (07 1995), 337–354.

[18] Ray, T. S. An approach to the synthesis of life. In *Artificial Life II* (Redwood City, CA, 1991), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds., Addison-Wesley, pp. 371–408.

[19] Salzberg, C., Antony, A., and Sayama, H. Evolutionary dynamics of cellular automata-based self-replicators in hostile environments. *Biosystems 78*, 1 (2004), 119–134.

[20] Sayama, H., and Nehaniv, C. L. Self-reproduction and evolution in cellular automata: 25 years after evoloops. *Artificial Life 31*, 1 (02 2025), 81–95.

[21] Sipper, M. Fifty years of research on self-replication: An overview. In *Artificial Life*, C. G. Langton, Ed. MIT Press, Cambridge, MA, 1998, pp. ??–??

[22] Taylor, T., and Dorin, A. *Rise of the self-replicators: Early visions of machines, AI and robots that can reproduce and evolve*. Springer Nature, 2020.

[23] Tempesti, G. A New Self-Reproducing Cellular Automaton Capable of Construction and Computation. In *Advances in Artificial Life (ECAL 1995)*, vol. 929 of *Lecture Notes in Computer Science*. Springer, 1995, pp. 555–563.

[24] Trevorrow, A., and Rokicki, T. Golly.

[25] von Neumann, J. *Theory of self-reproducing automata*. University of Illinois press Urbana, 1966, Editor: A.W. Burks.