To Evolve or Not to Evolve? That is the Question

Alex Ellery[1], A. E. Eiben[2]

[1]Department of Mechanical & Aerospace Engineering, Carleton University, 1125 Colonel By Drive, Ottawa, ON. K1S 5B6. Canada: aellery@mae.carleton.ca

[2]Department of Computer Science, Vrije Universiteit Amsterdam, de Boelelaan 1081a, 1081HV Amsterdam. Netherlands: a.e.eiben@vu.nl

Abstract

To evolve or not to evolve? That is the question: whether 'tis nobler in the mind to suffer the slings and arrows of grey goo, or to deny evolution to a sea of self-replicators and by prevention control them? We have been developing a physical self-replicating machine concept for deployment on the Moon built from local resources on the Moon. Here, we are concerned with architectural issues - we specifically address the problem of uncontrolled replication. We propose a multitiered approach to prevent this: (i) denial of self-replication through the implementation of centralised mass manufacturing of replicators; (ii) denial of scarce sodium and chlorine from Earth acts as an Earth-controlled kill switch in preventing further replication; (iii) denial of centralised supplies of asteroidal metals (tungsten-nickel-cobalt-selenium) at the lunar south pole acts as a Moon-controlled kill switch; (iv) denial of online learning capacity through fixed neural weights; (v) denial of extended computing resources through the elimination of transmit communications between self-replicators; (vi) denial of evolutionary capacity by implementing error detection and correction (EDAC) coding. Two kill switches and EDAC provide the backbone to our approach that maintain self-replication capability.

Introduction

We have been developing a physical instantiation of a self-replicating machine concept for service on the Moon to robotically construct a lunar infrastructure at low cost using local resources (Ellery, 2015a, 2016, 2017). To date, most effort has been devoted to 3D printing certain crucial components: (i) electric motors which has progressed to near completion; (ii) active computational components (vacuum tube) which has yet to be achieved but efforts are ongoing. We are also concerned with an important architectural issue – that of the prevention of uncontrolled replication. In approaching this problem, we are mindful of the Royal Navy's hard-learned lessons during the Falklands conflict regarding layered air defence for individual ships and flotillas, of which there are four – air combat patrol (Sea Harrier/F35 Lightning), area air defence (Sea Viper), point air defence (Sea Wolf) and close-in weapons (Phalanx/Goalkeeper). We explore a similar multi-tiered approach as our defence strategy against uncontrolled replication: (i) denial of self-replication through the implementation of centralised mass manufacturing of replicators; (ii) salt contingency – denial of scarce sodium and chlorine from Earth acts as an Earth-controlled kill switch in preventing further replication; (iii) tunicose contingency – denial of centralised supplies of asteroidal metals (tungsten-nickel-cobalt-selenium) at the lunar south pole acts as a Moon-controlled kill switch preventing further replication; (iv) denial of online learning capacity through fixed neural weights controls the machine's intelligence; (v) denial of extended computing resources through the elimination of transmit communications between self-replicators (receive only); (vi) denial of evolutionary capacity by implementing error detection and correction (EDAC) coding controls the machine's adaptability. We pay special attention to (i), (ii), (iii), (iv) and (vi).

3D Printer-Based Turing Machine – Denial of Online Learning

To address the problem of 3D printing computing machines, we revert to the original model of a computer. The Turing machine is a finite-state machine comprising a read/write head mounted onto an infinitely long tape divided into discrete squares. The Church-Turing thesis asserts that the mechanistic computations of a Turing machine define an algorithmic process. The Turing machine sequentially reads an infinitely long digital tape of cells. Symbols from a finite alphabet are inscribed on the tape which are read in sequence by the read/write head. The initial tape encodes a set of input data. The read/write head incorporates a finite memory of internal state transitions constituting the computer program of the Turing machine. The motion of the read/write head – the behaviour of the Turing machine - is determined by the symbol inscribed on each cell of the tape and the internal state of the machine. The symbol is overwritten by a replacement symbol and/or the read/write head moves one cell left or right according to the Turing machine's state transition function. The resulting tape encodes a set of output data. Different Turing machines are specified by different state transition functions. This simple machine implements a mathematical function that converts its input into an output – the Turing machine's mechanical procedure encapsulates the algorithm concept as a finite sequence of simple operations. Any specific Turing machine may be encoded as an input tape so a universal Turing machine can emulate any specific Turing

machine, i.e. a universal Turing machine can compute any computable function given the appropriate algorithm.

Our implementation of a Turing machine comprises an input tape represented by magnetic core memory, an output tape represented by an analogue neural net circuit, and a read/write head represented by a 3D printer. The 3D printer thus becomes a central component of a universal computation capability – it prints out hardware circuitry according to the program stored in magnetic core memory. Magnetic core memory uses ferrite magnetic cores (toroids) through which wires are passed to convey read and write signals. Each core stores one bit of information non-volatilely as zero or one depending on the direction of the core's magnetisation. The invention of the coincident current system enabled a small number of wires to control a large number of cores in 3D stacks. A large number of small ferrite toroidal cores are held on layers of XY grids of wires through the toroidal centres. Only where the combined magnetic field from X and Y lines cross exceeds a threshold will the magnetic polarity reverse. Magnetic core memory offers high reliability and was used for the Apollo Guidance Computer and Space Shuttle Flight Computers.

We must now consider the 3D printed output circuitry. We have adopted the vacuum tube as the basis of our active electronics. Vacuum tube devices are based on the generation of relativistic electron beams and their interaction with electromagnetic waves. A vacuum tube is simple in construction - a tungsten cathode that emits electrons attracted to a nickel anode controlled by a third nickel grid electrode encased in an evacuated glass or ceramic tube and linked by silicone or ceramic-insulated kovar wiring. Only a small number of materials are required which are readily extracted from lunar resources. However, vacuum tubes are bulky and present challenges for building complex computational circuits. The von Neumann architecture computer is based on the central processing unit (CPU). The core of the CPU is one or more arithmetic logic units (ALU). The ALU is a combinatorial logic circuit for performing arithmetic operations (addition, subtraction, increment/decrement and sign) and bitwise logical operations (AND, OR, EX-OR and NOT) on 4-bit, 8-bit, 16-bit, 32-bit or 64-bit data widths. For example, the modest embedded 8051 CPU comprises 2,200 logic gates. Modern computers comprise ~500 million logic gates. Data is stored in a variety of different memory locations which must be fetched as input data to the CPU and the results of which must be pushed back into memory. The basic operation of the von Neumann architecture is the fetch-decode-execute cycle which is wasteful in hardware footprint. Using vacuum tube-based circuitry based on the von Neumann architecture would require very large computers.

To prevent runaway growth in the computer footprint imposed by the vacuum tube, the output of our Turing machine is an analogue neural network that encodes a specific algorithm in hardware form. The complexity of a neural network increases only with the logarithm of the task complexity unlike the exponential increase in circuit complexity of digital architectures (Parberry, 1994). Neural networks are under development for general purpose intelligence - the SpiNNaker (spiking neural network architecture) project is based on combining a large number of digital ARM processors within a grid of switches to emulate a vast neural network representing a small brain of ~10^6 neurons. A simple electronic ring circuit of neurons has been proposed that emulates the neural processing function of the neocortex (Hahnloser et al, 2000). There is the prospect of implementing robust albeit simple behaviours neurally - one of the simplest biological neural networks in a non-aquatic free-living animal is that of the nematode worm *C elegans*: it comprises 959 cells in total as a hermaphrodite (of which 302 are neurons) or 1031 cells in total as a male (of which 381 are neurons) with approximately 5000 synapses. This potentially gives us a minimum neural network size though an engineered version might be subdivided into subnetworks of more modest dimension. On a much smaller scale, analogue neural circuits offer rapid computation with some biological fidelity in reducing specific energy consumption (energy/neuron) but at the cost of a fixed neural architecture.

We have adopted a modified version of the Yamashita-Nakamura neuron (Yamashita & Nakaruma, 2007) which comprises an input summing amplifier, an inverting amplifier (for a step function) and a comparator. The weights of each neuron are pre-trained offline to implement its desired behaviour. We have demonstrated a pre-trained two-neuron hardware circuit implementing a Braitenburg control architecture of BV2/BV3 class (Braitenburg, 1984) performing automatic obstacle avoidance on a simple desktop mobile robot. We have begun exploring the potential for augmenting hardware neurons with online learning circuitry (Larson & Ellery, 2015). There are several intriguing possibilities for learning circuitry (Winter & Widrow, 1988; Martinelli & Perfetti, 1991) but we assume that we do not implement such capabilities to prevent uncontrolled learning – nevertheless, we have the quandary of requiring weight adjustment to permit fine-tuning of analogue neural circuits to variations in physical manufacture against the denial of online learning to ensure that behavior is both known and controlled.

Centralised Manufacturing – Denial of Self-Replication Capacity and Kill Switches

The most aggressive approach to prevent runway replication is denial of self-replication of productive capacity. For this approach, we consider two options (centralized versus distributed production) across two dimensions (variability versus no variability), yielding four different cases (Table 1):

	Centralised Production	**Distributed Production**
Identical Copies	Conventional factory (e.g. six-sigma)	Self-replicators without evolutionary variation
Mutated Copies	EvoSphere	Self-replicators with evolutionary variation

Table 1. Replicator population options

The conventional factory employs mass production and has been employed since the Industrial Revolution for the

worldwide production of goods. In traditional factories, goods are produced in which variations are minimized. Before distribution, the quality of the goods are checked, and indeed, one of the hallmarks of quality is the so-called six-sigma quality control protocol.

Regarding terminology, self-reproduction may be regarded as an inaccurate form of self-replication that permits variation in offspring (Adams & Lipson, 2009) but we use the over-arching term self-replication here with or without evolution. The EvoSphere concept envisions an entire ecosystem of physically and behaviourally evolving robots in the physical environment (Eiben, 2015a). It comprises a birthing clinic, a nursery and a living arena. In the birth clinic, robotic machines are constructed from raw materials; in the nursery, they undergo an online learning phase for fine-tuning their behaviours to their bodies; and in the living arena, only successfully graduated robots that perform their desired function are selected and are permitted to replicate exactly. The EvoSphere imposes two apparently contradictory objectives: (a) reproduction with variation and selection is permitted to implement robot evolution in the real world; (b) a kill switch is implemented to prevent procreation of undesired variants by human operators. Robot reproduction is divided into two phases that differentiate between the robot genotype and the robot phenotype: (a) recombination of robotic genotypes is permitted without constraint; (b) production of the genotype-encoded robot phenotype is constrained. This constraint is imposed by permitting only a single centralised production centre for the physical construction of robots. The kill switch is implemented at the centralised production centre. If invoked, it shuts down all reproductive processes and, as a consequence, halts evolution. All prior variations of robot generations however are permitted to continue operation.

Versions of the kill switch through centralized production facilities have been proposed in approaches (ii) and (iii) to deny specific resources for self-replicators. In Fig 1, we present a lunar industrial ecosystem with recycling loops representing the required chemical processing to yield material feedstock for 3D printing of the self-replicator from lunar raw material. Self-replication requires precisely green chemistry (Anastas & Warner, 1998) in order to achieve the material closure implicit in an industrial ecology. A corollary of this is that an evolving and diverging population would be wasteful in physical resources unless the littered carcasses of failed evolutionary experiments were scavenged efficiently. For our lunar ecosystem, the loss of iron-nickel-cobalt alloy from asteroidal resources – which must be mined from special ore locations on the Moon – and the loss of NaCl imported from Earth due to its scarcity of the Moon effectively decimates the entire ecosystem. The loss of tunicose materials prevent the manufacture of ferrite magnets (and so motors, etc), tool steel, permalloy, kovar, thermionic cathodes and photosensitive elements (tunicose contingency). Loss of NaCl prevents the manufacture of AlNiCo magnets (and so motors, etc), photosensitive elements, silica for transparent glass, piezoelectric sensors, regolith binder, drilling mud, silicone plastics and oils, Metalysis FFC process anode and electrolyte regeneration (salt contingency). These two sets of kill switches – one on the Moon and the other on Earth – provide the last lines of defence to uncontrolled replication. An important proviso is that as the number of replicating units grows, central supply hubs become traffic bottlenecks. Although these contingencies are specific to the self-replicating machine proposed for the Moon (for instance, the salt contingency would not be possible on Mars), they illustrate the effectiveness of multiple resource-denial kill switches.

Lunar Ilmenite

$Fe^0 + H_2O$ or silicone oil in colloidal suspension → ferrofluidic sealing

$FeTiO_3 + H_2 \xrightarrow{1000^oC} TiO_2 + H_2O + Fe$ (Fe separated by liquation)

ilmenite $2H_2O \rightarrow 2H_2+O_2$ (H_2 recycling)

$2Fe + 1.5O_2 \rightarrow Fe_2O_3/Fe_2O_3.$**CoO** - **ferrite magnets**

Nickel-iron meteorites

Mond process:

		Alloy	Ni	Co	Si	C	W
$Fe(CO)_5$ ↔ 5CO + **Fe** (175°C/100 bar)	→	**Tool steel**				<2%	9-18%
$Ni(CO)_4$ ↔ 4CO + **Ni** (55°C/1 bar)	→	Electrical steel			3%		
$Co_2(CO)_8$ ↔ 8CO + **2Co** (150°C/35 bar)	→	**Permalloy**	80%				
		Kovar	29%	17%	0.2%	0.01%	.
W$(CO)_6$ ↔ 6CO + **W**	→	**Thermionic cathodic material**					

$4FeS + 7O_2 \xrightarrow{S\ catalyst} 2Fe_2O_3 + 4SO_2$

(Troilite) $SO_2 + H_2S \rightarrow 3S + H_2O$

FeSe + **Na_2**$CO_3 + 1.5O_2 \xrightarrow{KNO_3\ catalyst} FeO +$ **Na_2Se**$O_3 + CO_2$

Na_2Se $O_3 + H_2SO_4 \rightarrow$ **Na_2**$O + H_2SO_4 +$ **Se** → **photosensitive Se**

↑_____|

Na_2 $O + H_2O \rightarrow 2$**Na**OH (recycle)

Lunar Orthoclase

$3KAlSi_3O_8 + 2$**HCl** $+ 12H_2O \rightarrow KAl_3Si_3O_{10}(OH)_2 + 6H_4SiO_4 + 2KCl$

orthoclase illite silicic acid (soluble silica)

$2KAl_3Si_3O_{10}(OH)_2 + 2$**HCl** $+ 3H_2O \rightarrow 3Al_2Si_2O_5(OH)_4 + 2K$**Cl**

kaolinite (clay) → **porcelain**

$H_4SiO_4 \rightarrow SiO_2 + 2H_2O$

$KCl + NaNO_3 \rightarrow NaCl + KNO_3$ (recycle)
saltpetre

Olivine

$Mg_2SiO_4 + 2CH_4 \rightarrow 2CO + H_2 + 2MgO + Si$ at 2000°C → 3D Shaping binder

forsterite $MgO + HCl \rightarrow MgCl_2 + H_2O$ → **3D Shaping binder**

$2Mg + Si \rightarrow Mg_2Si$ → thermoelectric conversion

Lunar Anorthite

$CaAl_2SiO_8 + 4C \rightarrow CO + CaO + Al_2O_3 + 2Si$ at 1650°C → CaO cathode coatings

→ Portland cement

$CaO + H_2O \rightarrow Ca(OH)_2$

$Ca(OH)_2 + CO_2 \rightarrow CaCO_3 + H_2O$

$CaAl_2SiO_8 + 5HCl + H_2O \rightarrow CaCl_2 + 2AlCl_3.6H_2O + SiO_2$ → fused silica glass

→ **Metalysis FFC electrolyte**

$AlCl_3.6H_2O \rightarrow Al(OH)_3 + 3HCl + H_2O$ at 100°C

↑_____|

$Al(OH)_3 \rightarrow Al_2O_3 + 3H_2O$ at 400°C $\rightarrow 2Al + Fe_2O_3$ → $2Fe + Al_2O_3$ (thermite)

→ Al wiring

→ **AlNiCo hard magnets**

→ reflective surfaces, e.g. Al solar sail

Lunar Pyroxene

$CaFeSi_2O_6 + HCl + H_2O \rightarrow Ca_{0.33}(Al)_2(Si_4O_{10})(OH)_2.nH_2O + H_4SiO_4 + CaCl_2 + Fe(OH)_3$

pyroxene calcium montmorillonite soluble silica

→ **bentonite drilling mud/sand casting binder**

→ **Metalysis FFC electrolyte**

Lunar Volatiles

850°C 250°C

$CH_4 + H_2 \rightarrow CO + 3H_2 \rightarrow CH_3OH$ 350°C

Ni catalyst Al_2O_3 $CH_3OH + HCl \rightarrow CH_3Cl + H_2O$ 370°C $+nH_2O$

Al_2O_3 $CH_3Cl + Si \rightarrow (CH_3)_2SiCl_2 \rightarrow ((CH_3)_2SiO)_n + 2nHCl$

↑_____|

→ **silicone plastics/oils**

$3NO + H_2O \rightarrow 2HNO_3 + NO$

↑_____|

$2SO_2 + O_2 \leftrightarrow 2SO_3$ (low temp)

$SO_3 + H_2O \rightarrow H_2SO_4$

Metalysis FFC Process ($CaCl_2$ electrolyte)

$MO_x + xCa \rightarrow M + xCaO \rightarrow M + xCa + ½xO_2$ where M=Fe, Ti, Al, Mg, Si, etc, (O_2 consumes graphite anode)

$CO + 0.5\ O_2 \rightarrow CO_2$

$CO_2 + 4H_2 \rightarrow CH_4 + 2H_2O$ at 300°C (Sabatier reaction) $\rightarrow CH_4 \rightarrow C + 2H_2$ at 1400°C

Ni catalyst → **graphite anode regeneration**

Salt of the Earth

$NaCl + CaCO_3 \rightarrow Na_2CO_3 + CaCl_2$ → **Metalysis FFC electrolyte**

350°C/150 bar

$Na_2CO_3 + SiO_2(i) \leftrightarrow Na_2SiO_3 + CO_2$ → **piezoelectric quartz**

$NaCl + HNO_3 \rightarrow HCl + NaNO_3$

Fig 1. In-situ resource utilisation schematic with highlighted kill switches indicating loss of function

The full centralised replicator production facility approach denies the prospect of exponential growth of productive capacity (though the kill switches retain their applicability and effectiveness). A self-replicating machine offers unprecedented productive capacity by virtue of its exponential population P growth:

$$P = \sum_{i=1}^{m} (1+r)^i$$

where r=number of offspring per generation, i=generation number, m=number of generations. In a self-replicating machine, the number of offspring per generation, r≥1, e.g. our lunar application proposed r=2 and m=13 yielding P~2 x 10^6, i.e. the power of self-replication yields exponential growth in production. This is both its attraction and its prospect for jeopardy. Without restraint, robots can reproduce including their own reproductive capacity – this is equivalent to cell division in single cell populations, larvae in arthropods, egg laying in fish, reptiles and birds, and pregnancy in mammals. This introduces the potential for runaway replication with two specific dangers: (a) uncontrolled population growth (a macroscopic version of the grey goo scenario); (b)

uncontrolled and unwanted alterations in form and function of individuals. The dangers that this represents cannot be permitted (Eiben et al, 2012).

Error Detection & Correction Coding – Denial of Evolutionary Capacity

Despite the inadequacies of an operational definition of life to be a self-sustained chemical system capable of Darwinian evolution (Cleland & Chyba, 2002), we adopt it here with the corollary that any kind of self-replicator that copies genetic information is subject to Darwinian evolution due the genetic mutation (Ellery, 2018). Biological evolution has been broadly characterised by a growth in genetic complexity in that complex biological phenotypes require increased amounts of genetic information encoded from the environment in which they have evolved (Adami et al, 2000). Algorithmic complexity in a Kolmogorov sense may be regarded as the shortest bit sequence that can yield a given output. Although there are exceptions due to the C-value paradox (there is a fraction of non-coding genes that varies across species), genomic complexity reflects phenotypic complexity. At gene site i, there are four possible nucleotides with probabilities ($p_C(i)$,$p_G(i)$,$p_A(i)$,$p_T(i)$) yielding an entropy per site:

$$H(i) = -\sum_i^{j=4} p_i(i) log p_i(i)$$

Hence, the maximum entropy per site is two bits due to base pair complementarity of A-T and C-G. This permits computation of the physical complexity of the organism as a whole by:

$C = L - \sum_i H(i)$ where L=genome length (bp).

We expect this trend to continue in the self-replicating machine – evolutionary variation through generations implies a degree of uncontrollability. Our goal is to prevent evolutionary processes in the self-replicating machine to retain controllability. One proposal suggests that runaway self-replication of machines on Earth will yield at least 4°C temperature rise assuming thermal pollution within 2 years which would be readily detectable (Freitas, 2001). We aim to prevent uncontrolled replication in the first place so any self-replication scheme requires an error detection and correction (EDAC) coding strategy which we briefly review here (Griffith et al, 2005).

Biological self-replication employs template molecules to make copies of itself using building blocks in its environment. The invariant strands of alternating pentose and phosphate groups mount the four bases as rungs which form specific pairs A-T and G-C between purines (A and G) and pyramidines (C and T). DNA bases A, T, G and C form a digital code with three-quad codons such as GCC which encodes the amino acid alanine. This provides 64 codons for only 20 amino acids providing redundancy in the genetic code, e.g. UC* codes for serine so the third position of the codon is a wildcard. There are others however that are uniquely coded, e.g. UGG is the unique code to tryptophan.

In its normal packaging state, DNA is coiled up in chromatin proteins which unrolls the sticky DNA strings rapidly for copying and re-rolls it after copying. DNA helicase separates DNA into two strands in preparation for replication. DNA polymerase then creates two double stranded DNA strings from the two separated single strands. The average copying error rate in human DNA is ~10^{-8} but this varies with the sensitivity of the gene to allelic variation – histone genes which code for DNA packing proteins appear almost invariant to mutation across biological domains. The most significant treatment of information theory to genetics is the visionary monograph by Hubert Yockey that deserves wider recognition (Yockey, 1992).

The evolutionary pressure for genetic parsimony favours the short overlapping genes of low complexity eukaryotic viruses and DNA phages but favours non-overlapping modular genes in higher organisms (Ofria & Adami, 2002). Overlapping genes with multiple expression require slower evolutionary change. Neutral mutations occur in the third nucleotide of a codon afforded by coding redundancy but overlapping genes have offset reading frames making neutral mutations impossible. Larger genomes cannot exploit overlapping genes. The conundrum of replication copying fidelity requiring high genomic complexity (length) during early life before such high copying fidelity could evolve is referred to as the "error catastrophe" (Joyce, 2002). At higher mutation rates, genotypes of higher mutational robustness with lower replication rates are favoured irrespective of replication fidelity per genome $F=e^{-RL}$ where R=error rate per base pair, L=genome length (base pairs) (Wilke et al 2001). The vast majority of eukaryotic DNA is non-coding – in humans, only 3% of the genome is active, the other 97% being pseudogenes, etc. In eukaryotes, there is "junk" DNA (introns) that is excised from mRNA before being translated into proteins. Introns are marked for excision by a start sequence GT and a stop sequence AG after being looped to bring the active flanking genes into proximity and then cut out. Noncoding regions of the eukaryotic genome exhibit long range correlations (Buldyrev et al, 1995). There are epigenetic mechanisms for switching out genes - methylation of DNA either attaches methyl groups directly to DNA or modifies histones which dictate the activation status of genes.

In biology, the error correcting process during DNA replication involves proofreading (known as 3'-5' end exonuclease) (Battail, 2004). In DNA, error detection and correction is accomplished with polymerases which check the complementary fit between base pairs. In bacteria, the three DNA polymerases I, II and III progress along the growing dual DNA strand from the 5' end to the 3' end, recognise incorrect bases, reverse direction from the 3' end to the 5' end, excise the incorrectly matched base and re-insert the correct base. There are other repair mechanisms for base excision repair, mismatch repair, strand break repair, cell cycle checkpoints and cell apoptosis. There are many environmental disruptions that can occur in cells: UV radiation can cause cytosine and thymine to fuse distorting the DNA shape which effectively marks the region to be excised; deamination converts the GC base pair to an AT base pair which can be corrected by DNA glycosylases; oxidised guanine emulates thymine and must be replaced. These are point mutations. Deletions and insertions of single base pairs

generate frameshift mutations which cause entire reading frames to become shifted. Similar frameshifts occur when entire sections are deleted, copied or inverted. Transposons are sections of DNA that cut-and-paste themselves out of and into different locations. In each case, it is the complementary base that provides the reference datum, detectable helix distortion, diploid genome copies and predictable corruptions that permit repair. The repair mechanisms however are not perfect but species such as *Deinococcus radiodurans* can survive extreme radiation environments by using an average of four to ten copies of its genome (evolved to cope with extremely dry conditions) (Battista, 1997). This is a form of repetitive coding. Such a form of coding may be employed through multiple gene copies – triple (or higher order n-tuple) modular redundancy with voting logic is common in safety critical systems in spacecraft. An example of such a safety critical function would be a Hayflick limit on the number of generational copies that a self-replicating population can produce, e.g. emulating telomere shortening as a counter.

For our self-replicating machine, it is essential that codec (coding/decoding) can be performed using simple circuitry and/or neutrally using analogue neural network circuits. We assume that genetic information is transmitted vertically through generations and hierarchically through the population as it increases (Battail, 2010). This constitutes a noisy and bursty communications channel for the transmission of genetic information. The maximum information transmission rate through a communication channel is given by Shannon's coding theorem:

$R<B\log_2(1+SNR)$

where B=bandwidth and SNR=signal-to-noise ratio. One way to transmit error-tolerant messages is to transmit the message f times (repetition coding) – if f=3, triple modular redundancy permits a simple majority voting logic. This is highly inefficient. A more efficient way is to add parity bits to the message data. Any kind of channel code adds structured redundant bits increase the fidelity of information transmission at a cost of higher bandwidth requirements (Berlekamp, 1980). Typically, error detection and correction (EDAC) codes are usually implemented in hardware using extra memory bus bits and encoding/decoding circuits. The data lines of the EDAC bus connect directly to RAM. Address lines to the memories are buffered by latches which synchronise the address to the system clock allowing synchronous burst of instruction and data caches. The SNR can be related to normalized signal-to-noise ratio per bit E_b/N_0 by:

$E_b/N_0=(S/N)/(R/B)$

Above the theoretical Shannon coding limit, there is a code that can communicate with zero error (Costello & Forney, 2007):

$E_b/N_0>(2^{R/B}-1)/(R/B) = \ln 2 = -1.6$ dB

A typical bit error rate (BER) used in spacecraft communications is 10^{-6} (less stringent that the biological BER of 10^{-8}) but the BER will depend on the genome size for the self-replicator. A BER of 10^{-9} is routinely achievable in space systems and that a BER of 10^{-15} is desirable – with a maximum population limit of 10^6 self-replicating units, this gives a standard BER of 10^{-9} per machine. The closest to the Shannon coding limit achievable are turbo codes which are formed by the parallel concatenation of two recursive codes separated by an interleaver code (Berrou et al, 1993). The interleaver is the crucial aspect as it implements a pseudo-random code. Turbo-codes are complex and are unlikely candidates for biological implementation. The extensive tandem repeats and introns in the eukaryotic genome and especially the human genome may be implementing error detection and correction codes as parity bits. One of the simplest error detection and correction codes is the Hamming code such as the Hamming (7.4) code which can correct a single-bit error but detect one-bit and two-bit errors. It has been suggested that exon-intron genes are Hamming codewords (Faria et al, 2012). There are two main types of EDAC other than turbo codes – block codes and convolutional codes. In an (n,k) linear block code, there are k information bits (input block) and n-k parity bits for n message bits in total (output block) with a code rate of r=k/n (Bhargava, 1983). There are 2^k possible different messages of length k that are mapped onto 2^k codewords of length n. The summations require modulo-2 arithmetic without carries which can be implemented through the memoryless EX-OR circuit. The Hamming weight of a code word w(c) is the number of nonzero components to the code word. The code can correct any pattern of e or fewer random errors provided $2e+1\leq d$ where d=Hamming distance between two codewords (number of different elements in the codewords). The primary goal of block coding is to maximise the Hamming distance between codewords. The commonest used block codes are the Bose-Chaudhuri-Hocquenhem (BCH) codes and Reed-Solomon (RS) codes with the following parameters (Table 2):

	Block length	**Number of coding bits**	**Hamming distance**
Hamming code	$n=2^m-1$	k=m	d=3
BCH code	$n=2^m-1$ for m=3,4,5,…	$k\geq n-me$	$d\geq 2e+1$
RS code	$n=2^m-1$ symbols with m bits/symbol	(n-k)=2e symbols with m bits/symbol	d=2e+1 symbols

Table 2: Properties of some common block codes

The Reed-Solomon code is a type of non-binary BCH code. The Reed-Solomon code is of particular interest because it offers the maximum Hamming distance for an (n,k) code with Hamming distance d=n-k+1 and can correct bursts of e symbol (m-bit) errors per codeword (Berlekamp, 1982). This is because an m-bit burst is concentrated into a single symbol error. The BCH code is decoded by an iterative Berlekamp-Massey decoding algorithm (Imamura & Yoshida, 1987). There is evidence that block codes such as BCH codes appear

to be implemented in evolutionarily ancient gene sequences of the *Arabidopsis* brassica flowering plant (Brandao et al, 2015). The Golay code is a "perfect" three-error block code that works only for (23,12) for d=7 and (24,12) for d=8 – it is based on a remarkable number theoretic relation:

$$\frac{2^{23}}{\binom{23}{0}+\binom{23}{1}+\binom{23}{2}+\binom{23}{3}} = 2^{23}/2^{11} = 2^{12}$$

where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$=binomial coefficients. It would be curious indeed if evolution had discovered the tri-error Golay code for the tri-base genetic code.

A convolutional code of code rate 1/r is a type of trellis code which can be generated by a sequential k-stage shift register with r modulo-2 adders. Convolutional codes do not segment the information stream into blocks but add redundant bits continuously and so requires a memory of order m. Each branch in the decoding trellis is labelled with an n-bit output block, so there are 2^n branch metrics. A common coding protocol is a constraint length of k=7 and code rate r=1/2 for decoding efficiency. Convolutional codes are decoded using the Viterbi algorithm which is a maximum likelihood decoder (Forney, 1973). A shift register represents every state in the decoding trellis. The complexity of Viterbi decoders is exponential with the constraint length of the code. An artificial neural network Viterbi decoder based on analogue neurons has been demonstrated (Wang & Wicker, 1996). It implemented discrete connections weights (+1,-1) to eliminate the need for network training. Neurons represented trellis elements and selected the maximum trellis path at each pass and updated the path metrics through feedback connections. It was a locally connected network to minimize its complexity. 2^n neurons are required for an n-bit input while 2^m neurons are required for m encoding feedback connections (m=k-1). For each state, $2(2^k-1)$ neurons are required to find the maximum metric of the 2^k paths. Thus, the total number of neurons required is $n=2^{m+k+2}+2^n-2$. For a r=1/2, k=7 convolutional code, n=514 neurons offering a much smaller footprint than a digital ASIC. The neural Viterbi algorithm performed significantly faster than a digital implementation due to its parallel architecture.

The CCSDS (consultative committee for space data systems) standard for spacecraft recommends concatenation of two EDAC by interleaving an inner (7,1/2) convolutional code (applied first) with an outer Reed-Solomon (255,223) block code (applied last) for high data rate telemetry downlinks or BCH (63,56) for low data rate command uplinks. This is a specific example of Battail's nested code characterising aspects of the biological genetic code such as highly conserved HOX genes (Battail, 2008). HOX genes determine head-tail topological structure through morphological gradients and have diverged little since the emergence of multicellular organisms in the Cambrian explosion 540 My ago. The evolutionary rate can be controlled to significantly reduce evolutionary divergence. The CCSDS protocol illustrates that EDAC may be nested multiple times to give an arbitrary error rate, though of course at the cost of memory consumption. Resources devoted to EDAC during copying from n random components in the environment increases linearly as error rate decreases exponentially as $(1-e)^n$ (Griffith et al, 2005).

Conclusions

It appears that evolutionary divergence in a growing population of self-replicating machines are inevitable but prudence dictates that a multi-tiered system of safeguards should be adopted. The first layer of defence is the implementation of costly modular redundancy (say, five gene copies emulating the five modular redundancy of the integrated computers onboard the Space Shuttle) and multiple recursive layers of EDAC to reduce evolutionary divergence. The second layer of defence is the prevention of online learning through fixed weight neural networks – related to this is minimisation of social interaction through fixed communication protocols (which we have not addressed here). The third line of defence constitutes two layers of kill switches that are self-replicating machine specific – on the Moon, controlling access to centrally mined tungsten-nickel-cobalt-selenium from asteroidal resources, and on the Earth, denial of the reagents sourced from NaCl that must be transported from Earth. The final, most aggressive proposal is centralisation of all mass production facilities but this drastically reduces the attractive aspects of self-replication so we do not consider this to be practical.

A more philosophic issue is that by implementing EDAC we are effectively halting the evolutionary process by introducing high levels of copying fidelity. If life is defined as a self-sustained chemical system capable of undergoing Darwinian evolution (Luisi, 1998), there are three plausible interpretations of this definition. The first is that by denying evolutionary processes, our self-replicator is no longer alive as it is no longer subject to evolutionary processes. The second is that we have only suppressed the evolutionary process but not the capacity for evolutionary development, so it is alive. The third is that this evolutionary suppression is not absolute but based on bit error rate – later rather than sooner, copying errors will arise if the population grows beyond any imposed Hayflick limit, i.e. we have slowed evolution rather than halted it. Of course, the first interpretation states that this is not possible because of the integrity of the Hayflick limit imposed by EDAC. We have come full circle… A similar situation occurs in attempts to suppress evolution in synthetic biological organisms (Schark, 2012).

Experiments in Avida indicate that self-replication does not guarantee evolvability (LaBar et al, 2015). Within Avida, a lack of evolvability can occur if all possible mutations to a specific genetic sequence prevent further self-replication. The closest biological organism in which this occurs is the mule which is a horse-donkey chimera that is nominally sterile; it might occur in engineered systems in which evolutionary brittleness is a consequence of the genetic encoding system – this has been explored in genetic algorithms to reduce brittleness through cellular encoding of embryonic development in hardware systems (Eiben & Smith, 2015b) such as genetic programming trees (Funes & Pollack, 1998) or L-systems. These represent growth processes which are tolerant of mutations but direct encodings are much more brittle rendering the possibility that engineered self-replicators may be designed so that they always mutate into dysfunction (most mutations are in fact dysfunctional – here, all mutations would be so).

Another interesting option concerns whether learning capability and evolutionary capacity might be permitted but moderated. This introduces the notion of shaping the learning or evolutionary process. In neural networks, this requires initialization of the network weights to incorporate pre-defined structures prior to learning (innate knowledge). Symbolic connectionism incorporates expert system-based structures into neural networks (Ellery, 2015b). Bayesian networks are particularly suitable as a priori neural network knowledge which imposes structure to any subsequent learning. In genetic algorithms, such shaping is imposed through the fitness function – usually a simple metric, there is no reason why it cannot become more prescriptive – indeed, the design of a planetary rover's chassis (number of wheels, wheel radius and width, grouser size, vehicle weight, etc) was successfully evolved using a fitness function that implemented maximization of drawbar pull computed through a Bekker-Wong terramechanics model (Setterfield & Ellery, 2010 *unpublished data*). In effect, the fitness function substituted for the environment. The question is how to implement such fitness functions into a self-replicator to control the direction of its evolution.

These issues are, as yet, unexplored but warrant further investigation. The chief concern must be to what extent learning and evolution can be shaped and controlled. We are skeptical – until the advent of further evidence - that full control can be exerted because partial control is no control – indeed, partial control is more dangerous than no control because it offers an illusion of control where there is none.

References

Adami C, Ofria C, Collier T (2000) "Evolution of biological complexity" *Proc National Academy Sciences* **97**, 4463-4468

Adams B, Lipson H (2009) "Universal framework for analysis of self-replication phenomena" *Entropy* **11**, 295-325

Anastas, P, Warner J (1998) Green Chemistry: Theory and Practice, Oxford University Press: New York

Battail G (2004) "Engineer's view on genetic information and biological evolution" *BioSystems* **76**, 279-290

Battail G (2008) "Information theory and error-correcting codes in genetics and biological evolution" in *Introduction to Biosemiotics* (ed. Barbieri M), Springer Publishers, 299-345

Battail G (2010) "Heredity as an encoded communication process" *IEEE Trans Information Theory* **56** (2), 678-687

Battista J (1997) "Against all odds: the survival strategies of *Deinococcus radiodurans*" *Annual Reviews Microbiology* **51**, 203-224

Berlekamp E (1980) "Technology of error-correcting codes" *Proc IEEE* **68** (5), 564-593

Berlekamp E (1982) "Bit-serial Reed-Solomon encoders" *IEEE Trans Information Theory* **28** (6), 869-874

Berrou C, Glavieux A, Thitmajshima P (1993) "Near Shannon limit error-correcting coding and decoding: turbo-codes" *Proc IEEE Int Conf Communications*, paper no 397441

Bhargava V (1983) "Forward error correction schemes for digital communications" *IEEE Communications Magazine* **21** (1), 11-19

Braitenburg V (1984) "Vehicles: Experiments in Synthetic Psychology" *MIT Press*

Brandao M, Spoladore L, Faria L, Rocha A, Silva-Filho M, Palazzo R (2015) "Ancient DNA sequence revealed by error-correcting codes" *Scientific Reports* **5**, 12051

Buldyrev S, Goldberger A, Havlin S, Mantegna R, Matsa M, Peng C-K, Simons M, Stanley H (1995) "Long-range correlation properties of coding and noncoding DNA sequences: GenBank analysis" *Physical Review E* **51** (3), 5084-5091

Cleland C, Chyba C (2002) "Defining life" *Origins of Life & Evolution of the Biosphere* **32**, 387-393

Costello D, Forney D (2007) "Channel coding: the road to channel capacity" *Proc IEEE* **95** (6), 1150-1177

Eiben A (2015a) "EvoSphere: the world of robot evolution" *Proc Theory & Practice of Natural Computing* LNCS 9477 (ed. Dediu A-H et al), Springer Publishers, 1-17

Eiben A, Smith J (2015b) "From evolutionary computation to the evolution of things" *Nature* **521**, 476-482

Eiben A, Kernback S, Haasdijk E (2012) "Embodied artificial evolution: artificial evolutionary systems in the 21st century" *Evolutionary Intelligence* **5** (4), 261-272

Ellery A (2015a) "Engineering artificial extraterrestrial life?" *Proc European Conf on Artificial Life (Late Breaking)*, York, UK, 12-14

Ellery A (2015b) "Artificial intelligence through symbolic connectionism – a biomimetic rapprochement" in *Biomimetic Technologies: Principles & Applications* (ed. Ngo D), Elsevier Publishing

Ellery A (2016) "Are self-replicating machines feasible?" *AIAA J Spacecraft & Rockets* **53** (2), 317-327

Ellery A (2017) "Building physical self-replicating machines" *Proc European Conf on Artificial Life*, Lyon, France, 146-153

Ellery A (2018) "Engineering a lunar photolithoautotroph to thrive on the Moon – life or simulacrum?" *Int J Astrobiology* S1473550417000532

Faria L, Rocha A, Kleinschmidt J, Silva-Filho M, Bim E, Herai R, Yamagishi M, Palazzo R (2012) "Is a genome a codeword of an error-correcting code?" *PLoS One* **7** (5), e36644

Forney D (1973) "Viterbi algorithm" *Proc IEEE* **61** (3), 268-278

Freitas R (2001) "Some limits to global ecophagy by biovirus nanoreplicators with public policy recommendations" *reprint*

Funes P, Pollack J (1998) "Evolutionary body building: adaptive physical designs for robots" *Artificial Life* **4** (4), 337-357

Griffith S, Goldwater D, Jacobson J (2005) "Self-replication from random parts" *Nature* **437**, 636

Hahnloser R, Sarpeshkar R, Mahowald M, Douglas R, Seung S (2000) "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit" *Nature* **405**, 947

Imamura K & Yoshida W (1987) "Simple derivation of the Berlekamp-Massey algorithm and some applications" *IEEE Trans Information Theory* **33** (1), 146-150

Joyce G (2002) "Booting up life" *Nature* **420**, 278-279

LaBar T, Adami C, Hintze A (2015) "Does self-replication imply evolvability?" *Proc 13th European Conf Artificial Life*, 596-602

Larson S & Ellery A (2015) "Trainable analogue neural network with application to lunar in-situ resource utilisation" *Proc Int Astronautics Federation Congress,* Jerusalem, IAC-15-D3.3.6

Luisi P (1998) "About various definitions of life" *Origins of Life & Evolution of Biospheres* **28**, 613-622

Martinelli G & Perfetti R (1991) "Circuit theoretic approach to the backpropagation learning algorithm" *IEEE Int Symp Circuits & Systems* **3**, 1481-1484

Ofria C, Adami C (2002) "Evolution of genetic organisation in digital organisms" in *Evolution as Computation* (ed. Landwehr L, Winfree E), Springer Publishing, 296-313

Parberry I (1994) *Circuit Complexity and Neural Networks*, MIT Press Foundations of Computing, Cambridge, MA

Schark M (2012) "Synthetic biology and the distinction between organisms and machines" *Environmental Values* **21**, 19-41

Wang X & Wicker S (1996) "Artificial neural net Viterbi decoder" *IEEE Trans Communications* **44** (2), 165-171

Wilke C, Wang L, Ofria C, Lenski R, Adami C (2001) "Evolution of digital organisms at high mutation rates lead to survival of the flattest" *Nature* **412**, 331-333

Winter R & Widrow B (1988) "Madaline II: a training algorithm for neural networks" *IEEE Int Conf Neural Networks,* 401-408

Yamashita Y, Nakamura Y (2007) "Neuron circuit model with smooth nonlinear output function" *Proc Int Symp Nonlinear Theory & its Applications*, Vancouver, 11-14

Yockey H (1992) *Information Theory & Molecular Biology*, Cambridge University Press