2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Theory of Evolutionary Systems Engineering

Simon Hickinbotham*, Rahul Dubey*, Edgar Buchanan*,
Imelda Friel†, Andrew Colligan†, Mark Price†, and Andy M. Tyrrell*
**Intelligent Systems & Robotics Research Group, School of Physics, Engineering and Technology*
University of York, UK
†*School of Mechanical & Aerospace Engineering, Queen's University Belfast*
Email: *simon.hickinbotham, rahul.dubey, edgar.buchanan, andy.tyrrell(@york.ac.uk),
†I.Friel, A.Colligan, M.Price(@qub.ac.uk)

***Abstract*—Evolutionary approaches to engineering design involve generating populations of candidate solutions that compete via a selection process iteratively, to improve measures of performance over many generations. Although the attractive properties of biological evolutionary systems have motivated researchers to investigate emulating them for engineering design, there has been an emphasis on using encodings of the technical artefacts themselves, rather than encoding a complete bio-inspired system which is capable of producing such artefacts. It is the latter approach which is the subject of this contribution: how might a bio-inspired system be designed that self-organises the process of engineering design and manufacture? To make progress in the application of evolutionary processes to problems in engineering design, the evolutionary model must encompass the complexity of systems engineering. A new theory of evolutionary systems engineering is presented, based on von Neumann's Universal Constructor Architecture (UCA), drawing from more recent understanding of biology and applying the resulting system to the task of engineering design. It demonstrates how individual bioinspired algorithms fit into a coherent whole, and how they can be combined to drive open-endedness in automated design. The resulting system provides a common language for multidisciplinary applications in generative design, whereby industrial systems engineering approaches can be developed using principles from the UCA for the first time.**

***Index Terms*—Evolvable Systems; Bio-inspired; Multi-Agent System**

I. Introduction

PRESENTED with a particular design challenge, a concept-level human designer is commonly thought to be free to suggest the best solution that comes to their mind. For example, in a simple bridge-construction problem, such an unconstrained designer might suggest a ferry or rope bridge instead of a Warren truss bridge structure. However, under the constraints of the manufacturing organisation, as described by Systems Engineering (SE), the full range of design options is increasingly restricted, leaving a particular design vulnerable to rival approaches. The SE 'V' diagram [1], [2] is funnel-shaped for a reason: it forces concept-level designs down to a small number of possibilities. Both traditional model-based systems engineering and traditional evolutionary algorithms are constrained to optimise within very tight bounds. How could a system escape these limits in engineering, as biology does via evolution?

This work was funded by EPSRC, UK, grant reference EP/V007335/1 "RIED: Re-Imagining Engineering Design"

Capturing design intent in Systems Engineering

Engineering design is the process of making operational products from abstract ideas. This involves obtaining one or more precise geometric representations of the product [3]. Straight away the dual nature of designed artefacts is revealed, in that the intention of a design (the *design intent*) must be manifested in a physical object, but cannot be encoded into the object – it is not possible to recover the intent from the physical object without external knowledge. In the simple example of a hammer, the design intent is to deliver momentum and energy into nail head, but this information cannot be discovered within the hammer itself. Biological artefacts also have this dual nature, but the absence of the designer in evolution makes this point moot. In industry, this problem is reduced by keeping a record of the decisions made during the design process. However, such record-keeping is not actionable. In addition, the geometry of the design must be translated into a physical object via *manufacture*, adding a third level of complexity to the artefact. So far, the design process must deal with intent, geometry and manufacture. Coupling these perspectives with multiple geometric representations needed for a single object, the imperative for Systems Engineering as a framework for managing the complexity of these processes becomes clear.

Systems Engineering (SE) is a discipline which attempts to align the specification, design and manufacturing of engineering artefacts with their intended use cases [4]: the set of functions needed to fulfil the requirements of the product or system. The way in which these processes happen is loosely analogous to biological phenomena and it is not unusual to encounter comparisons with the multi-scale levels of organisation in cells, tissues, organs, organisms, communities and ecologies in the natural world – see for example [5] and [6] figure 1. The SE process attempts to preserve the design intent through the design and production processes (via the current V diagram), but the design solution that comes out tends to be highly constrained. One of the attractions of approaching this via an evolutionary system is that a more unconstrained set of solutions are allowed to appear: diversity is key, and commutability and associativity across sub-systems is also key. Accordingly, there have been various attempts to use bio-inspired control systems to re-imagine how engineering design might be realised, e.g. [7], [8], with the goal of emulating the capabilities of biological systems to *self*-organise in an apparently efficient manner.

Although the attractive properties of biological systems have motivated researchers to investigate emulating them for engineering design, there has been an emphasis on using encodings of the technical artefacts themselves, rather than encoding a complete bio-inspired system which is capable of producing such artefacts. It is the latter approach which is the subject of this contribution: how might a bio-inspired system be designed that self-organises the process of engineering design and manufacture?

Bio-inpsired Systems Engineering

The emergence and evolution of life on earth is documented to have undergone a series of transitions, in which the scale of organisation and complexity has generally increased [9]. Natural evolution is apparently open-ended in its complexity [10], and it is capable of self-organisation via properties of self-reference, from simple feedback loops up to levels of recursion seen in the notion of semantic closure [11]. All of these properties are desirable in systems engineering because they offer a route to automating the response to new design applications and opportunities, but the challenge is to arrive at a formulation of the biological processes that is simple enough to capture the dynamics *but no simpler*, and all this despite any obvious *intent* in the (physical manifestation of the) system.

There exists in the literature many and varied computational methods of leveraging bio-inspired evolutionary and neural techniques among others to perform various tasks, for example in pattern recognition and optimisation [12]. The process of engineering design is resistant to direct evolutionary approaches because the space of possible designs is extremely large, with non-linear relationships between the specification of the engineered artefact and its performance criteria. More recently, a body of work [13]–[15], is emerging around the topic of generative systems, which are tasked with the challenge of producing artefacts from relatively simple and indirect encodings of growth patterns, describing and exploiting iterative growth processes in order to achieve the final form. However, the manufacturing process tends to be at most a minor consideration, going against many of the principles of systems engineering. There is no overarching framework for designing such bio-inspired systems. Is there more that can be done to exploit commonalities between design challenges, embrace new technologies as they emerge, and build new design methods based on the generative, open-ended power of evolution as observed in the natural world?

In this contribution, such a framework is presented as a means of obtaining the desirable properties outlined above to create a new systems engineering paradigm. The framework's power in covering such systems and the logic behind its organisation offers a route to unifying the range of concepts needed to achieve truly bio-inspired systems engineering. The framework is based on John von Neumann's Universal Constructor architecture [16] (UCA). The goal of von Neumann's design was to devise a machine that was capable of constructing a second machine that was more complex than itself. This was seen by von Neumann as the central problem which biological evolution had solved. The work was divided into two parts: a theoretical component, which describes how the task of self-reproduction could be broken down in to a set of sub-tasks; and an implementation of the theory in what is now known as a cellular automaton. Research on the UCA has tended to focus on the implementation of the machine - in the original cellular automata form, and emulated in Tierra and Stringmol [17]–[19]. These systems prove that the UCA can be implemented for self-replication, but the challenges beyond that for systems engineering are two-fold: firstly, such systems tend to focus on the problem of self-replication, but not on the application of self-replication to solving a particular design problem via evolution; secondly, the requirement for the system to increase in complexity has received relatively little attention, particularly with respect to how variation in the machine can be generated. The next section reviews this architecture and gives an interpretation that applies to biological and engineering systems simultaneously.

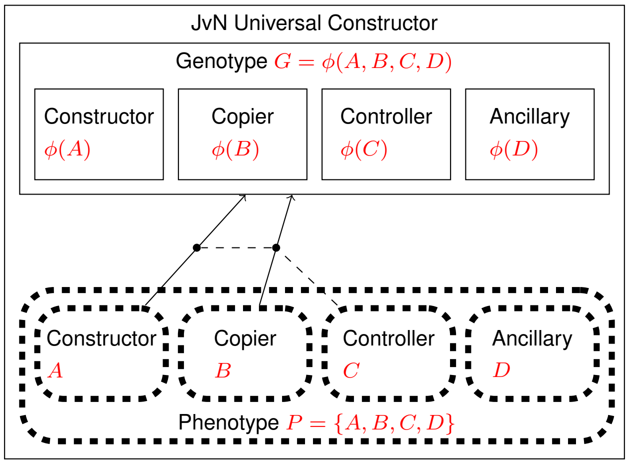

Fig. 1. John von Neumann's Universal Constructor Architecture. The abstract description, or *Genotype* is shown as boxes with solid lines. The executing machine, or *Phenotype* is shown as boxes with rounded corners and dashed lines. The Copier and Constructor machines act on the Genotype, as indicated by arrows.

II. Components of the Universal Constructor

A representative diagram of von Neumann's Universal Constructor (UC) is shown in figure 1. In an attempt to bridge the gap between von Neumann's description and the terminology of modern biology, the groupings 'Genotype' and 'Phenotype' are used to delimit the *description* of a component from the *instantiaion* respectively. In this aspect of his work, von Neumann was grappling with the problem of how the complexity of any system can increase. The system itself was not specified in detail, but mechanical artefacts were useful examples of such a system. A key insight was that a system of any complexity could not reproduce itself by self-inspection, since this would entail a process of (self-)disassembly in order to inspect the parts, and this in turn would prevent the system from functioning. What was needed was a simpler representation of the system, referred to here as the "abstract description" or the "genotype" to emphasise the biological analogy. This representation should describe the complete instantiation of the system, but its nature should be such that it would be relatively simple to i) copy the genotype; and ii) interpret the genotype to generate instances of the other entities in the system. Following this reasoning, a UC architecture (UCA) must contain the following components:

The Constructor: A

The role of the constructor A is to build a copy of all of the parts of the architecture as specified in the abstract description, including itself (but excluding the abstract description). Each component of the machine is less complex than the constructor, but the *ensemble* of machines is more complex, and is capable of becoming even more complex through evolutionary processes.

The biological analogy is that of the ribosome/tRNA complex, which aligns itself on mRNA strands and "decodes" the RNA sequence into a sequence of amino acids. Key to the success of this operation is the physico-chemical state of the cell in which the construction operation happens, which governs the folding of the 1-dimensional string of amino acids into the 3D shape of the artefact being constructed. If a cell is struggling to maintain benign internal conditions, the enzymes involved in reproduction are de-natured and the mutation rate increases, increasing the ability of the system to discover novel solutions. Thus, the "meaning" of a sub-assembly is partly embedded in the abstract description G, but also partly embedded in the prevailing physical conditions in which the decoding happens.

The Copier: B

The Copier's (B) sole responsibility is to create a copy of one thing: the abstract description of the machine – analogues of which include the 'genotype' or the 'blueprint' or possibly the CAD (computer-aided design) model of a designed artefact. Von Neumann recognised that a direct copying activity of some sort was vital for the UCA to function, but also that copying *descriptions* of things decouples the need to align the way things are allowed to change with the features that are used to represent them. Von Neumann also postulated that any entity which copies must be more complex than the entities which are copied by it, and solves the problem of how complexity can therefore increase by restricting what is possible to be copied and making the copier part of a wider assembly.

The copier embodies the evolutionary process of variation by introducing changes to the specification whilst creating the copy of it, commonly known as *mutation*. Epigenetic processes controlling the rate and location of mutations in the new copy do so by influencing the precision of the way the copier interprets the genome.

The Controller: C

The controller C is the component of the UCA that orchestrates the operation of the other replication machinery. In biological systems, the sort of control described here tends to be distributed, based on states induced by the gene regulatory network (GRN) among other control networks such as neural and immune systems. All these are interconnected, working at different spatial and temporal scales, and have similar properties of emergent control based on communication between networks.

The process of evolution allows appropriate control mechanisms to emerge through selection: individuals in the population with better control are selected for reproduction according to their fitness. Where the UCA is to be used for engineering purposes, the control mechanism can be approximated initially, and then optimised through evolution.

The Ancillary: D

Von Neumann was concerned with the process of replication in the UCA, so the focus of reasoning is upon this process. He recognised that there needs to be something to replicate beyond the replication machinery itself - and this is what the Ancillary machine D is. In biology, this can be any part of the phenotype that is not directly concerned with reproduction - the peacock's tail for example (NB survival to reproduce is not considered here). In engineering terms, 'product' or 'artefact' are appropriate labels.

The distinction between an engineering artefact and a living organism is clear in the UCA model: a biological organism (e.g., a Merlin *Falco columbarius*, the UK's smallest bird of prey) carries within it all the components it needs to manufacture a copy of itself (albeit in tandem with a reproductive partner). Instances of UCA components A, B, C, D and G can be identified in every living entity. By contrast, an engineering artefact such as the Rolls Royce Merlin engine [20], (One of the most successful aircraft engines of the 1940s) is not capable of reproduction - it is an instance of machine D in the wider context of an industrial manufacturing process, which contains instances of each of the other components in the UCA.

The Genome: G

The final component of the UCA is the abstract description of everything else in the system: $G = \phi(A, B, C, D)$. The description is one-dimensional - a string of symbols, which allows the work of copying it to be set out relatively simply, but also allowing evolutionary changes in the abstract system to emerge via various imprecisions in the way that copying takes place.

Properties of the UCA

The UCA solves the central problem of how complexity in a self-replicating system could increase. However, there are two particular features of the UCA that von Neumann recognised as problematic. Firstly, an evolving system implies variation in the abstract description of the machines. Whilst this variability is to be encouraged in machine D, it is far less likely to allow a replicating system to remain viable if mutations occur in A, B or C. These concerns can be mitigated by two biological phenomena. Firstly, it is now recognised that mutation rates vary widely across the genome [21]; and secondly the survival of *populations* of entities mitigates somewhat the potential catastrophic effect of deleterious changes in machines A, B and C in one particular offspring.

The second feature of the UCA system regards the requirement to encode the phenotype on the genome. Any process of encoding a specification of a system introduces notions of syntax and semantics, which involves the concept of *meaning* in the encoding. An idealised genome system could represent any physical design, but that does not happen, even in biology. Von Neumann wrote "by axiomatizing automata in this manner, one has thrown half of the problem out of the window, and it may be the more important half". However, the concept of *semantic closure* in these symbol systems, as explored by Pattee [11] and evaluated experimentally in [19], demonstrates that it is possible for the mapping of the genotype to the phenotype to evolve, as long as the Constructor machine B is itself encoded on the genome. This also provides a mechanism to tune the abstract representation such that viable or improved designs are more likely to emerge via evolution.

This process of tuning the representation has analogues in Systems Engineering, where for example a CAD model is parameterised in such a way that the manner in which it can be varied is aligned to its intended function [4], [22]. One of the attractive features of being able to evolve a representation is that the design intent of an artefact can be reflected in the way that artefact is represented. Thus, the UCA provides a mechanism to *evolve intent into the system*.

III. Partial implementations

Here, the way that two classes of bio-inspired systems fit into the UCA are described in order to draw out the limitations of such systems and how they relate to one another within this framework. The way these systems differ is in which components are represented on the genome, and are thus available to evolution. These are shown side-by-side in figure 2(a) and (b).

Evolutionary algorithms: $G = \phi(D)$ and fig. 2(a)

An abstract description of a machine X is denoted $\phi(X)$. Within the context of the UCA, canonical evolutionary algorithms only concern themselves with encodings $\phi(D)$ of machine D, as shown in figure 2(a). In these systems, it is only possible to optimize the system by variation in the set of parameters that are encoded on $G = \phi(D)$. Because the way that $\phi(D)$ is decoded into D by machine B is fixed, and because the way that changes on $\phi(D)$ produce variation is also fixed, the range of possible states of D is severely limited compared with the range of potential solutions in an unconstrained design space. For example, Deb and Gulati [23] evolve configurations of truss structures using a fixed set of nodes, where the genome represents the presence or absence of a connection between nodes. There is no possibility of moving any of the node positions to improve the structure - only the connections between them. Specifically, the complexity of D cannot increase. For many optimisation problems, this is sufficient, but for generative problems where a wider range of candidate designs is desirable, the arrangement may be too limiting.

Generative algorithms: $G = \phi(C)$ and fig. 2(b)

A generative system can be developed within the UCA model as having only the abstract description of the Controller C to the genome: $G = \phi(C)$. Unpicking this, it can be seen that the gene regulatory network architecture (GRN) [24] emerges from this situation: the interaction of C with A is applied iteratively to the phenotype D to allow it to grow into its final form, and the process controlling the growth of the organism is the only thing that can evolve. For example Hickinbotham et al [25] evolved a GRN for modifying the design of bridge trusses, allowing the position of nodes to move freely via an iterative development scheme using local feedback from finite element analysis.

IV. Extending bio-inspired systems into the UCA

The emphasis in the previous section has been on optimisation, either of the physical artefact as represented by D or of the control architecture for growing it, as represented by C. Where C is fixed, the system is identical to a evolutionary algorithm with a direct description of D, and because the description has to be direct, the resulting artefacts must be simple enough to discover with evolutionary search. Where D is fixed, the only thing that can evolve is the way that it grows, but growth always has to start with the same, simple version of D, potentially limiting what is possible given available resources. In addition, fitness measures as applied to the final form of the artefact play no *direct* part in organising the initial layout, or *body plan* – some additional components (seen in biology as the homeobox genes) are responsible for evolving the way that the body plan organises growth before the final structure can be tested. In this section, the effects of adding more abstract descriptions into G of the components of the UCA are explored. These are shown side-by-side in figure 2(c) and (d).

Homeobox EvoDevo algorithms: $G = \phi(C, D)$ and fig. 2(c)

If both the artefact D and the control mechanism C are encoded on G to give $G = \phi(C, D)$ as shown in figure 2(c), a much more powerful algorithm is possible: the two components of the UCA are able to combine to control the response to local conditions (via C) by reference to a *body plan* via D. The latter have been identified in biology as homeobox genes, whose contribution to the final shape of D is explored in [26].

The organisation of the development of D within a growth model organised by C can be interpreted as a production of a grammar from a start symbol. Shape grammars can be used to generate artefacts by stochastic firing of production

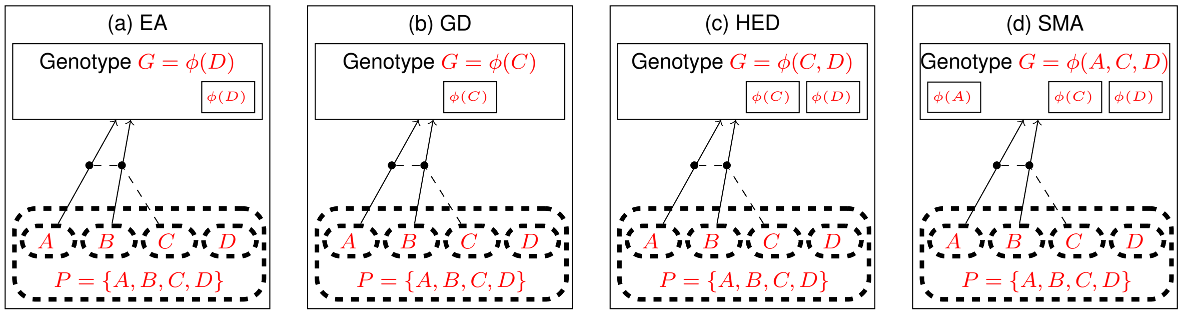

Fig. 2. Genetic Sub-components of the UCA. All the phenotye sub-components are present, but only a subset have genetic representations and are evolvable. These yield different bioinspired algorithms: a) Evolutionary Algorithm; b) Generative Design; c) Homeobox EvoDevo, d) Systems Manufacturing Algorithm

rules which have *attributes*, which allow a symbol to be interpreted as a shape. A comprehensive summary of the range of grammars available can be found in [27]. A key advantage of this approach is that the linkage between functional differentiation promoted by the body plan and growth promoted by feedback of local physical states can be combined to generate appropriate artefacts.

Systems Manufacturing algorithms: $G = \phi(A, C, D)$ *and fig. 2(d)*

Incorporating the specification of A on the genome has the effect of bringing (elements of) the manufacturing process under evolutionary control, allowing co-evolution of artefact and manufacturing process. Rather than directly using fitness measures for manufacturing on the form of D alone, evolvable practices could be encoded on the construction of D, as shown figure 2(d), then a better fit between A and D could be discovered via evolutionary search, allowing more efficient manufacturing methods to be discovered and optimised. This is a key facet of the UCA that realises the goal of allowing the complexity of designs to increase, compared with alternative approaches which fix features of systems forever by not including them on the genome.

V. Conclusion

As Artifical Intelligence systems get more complex in order to exploit the greater level of available compute power, the requirement to organise such systems increases in importance. The similarities between learning algorithms (which use training data to allow a model to reason over new data in some way) and generative evolutionary algorithms (which produce patterns without training data that are then subject to fitness measures) are driving the blurring of boundaries between these concepts. Common to all these endeavours is the question of how any generative system can increase its complexity in an appropriate manner to the task at hand. The Universal Constructor Architecture solves this central problem: it provides a framework for a complete evolutionary system that is able to increase its complexity by reference to a complete representation *of itself* on the genome. In addition, the UCA model provides a common language in which bio-inspired concepts such as evolutionary and generative algorithms are placed in context with the networks that control them.

Any observed intent in biology is *emergent:* organisms act in order to survive to reproduce and all other behaviours and properties are tailored in some way to meet that goal. This is the fundamental challenge in adopting bio-inspired techniques for engineering – how to encode intent within a system that does not capture intent explicitly. How then does this relate to the engineering challenge of design? Simply put, technical artefacts *also* have no intrinsic notion of intent: the purpose for which they were designed is external to the physical realisation of the design, and even the precise purpose of an artefact will vary in the minds of all parties involved with it. Kroess et al. [3] discuss the dual nature of technical artefacts, in which the relationship between the intention of the design and the artefact is described via the physical realisation of the physical aspects of design intent. Human intentionality is built into an artefact as it is produced, endowing technical artefacts with ‘for-ness’. The *function* of an artefact bridges its intentional and physical domains.

In computing, reflection is defined as any part of a computer program that is ‘about itself’ [28]. There has to be a *causal connection* between the source code and the executing program. If it is considered that the relationship between source and executing code is closely analogous to the relationship between genotype and phenotype, then von Neumann’s UCA embodies this causal connection, both in the constructor machine A and the control machine C. The former reads the genotype to make the machines in the phenotype, whereas the latter uses local feedback to control the growth of the artefact into its final form. In addition, the control machine is also evolved – providing a second level of causal connection directly related to fitness of the gestalt, which contributes to the goal of realising the “Evolution of Things” as described in [29]. These feedback loops, combined with the evolvability of the genotype-phenotype mapping, allow the UCA architecture the flexibility to change the way the system is represented to itself – a key requirement if intent is to be represented within the system, and a highly desirable property of systems engineering.

The novelty of this contribution has been to apply von Neumann’s UCA model to the modern paradigm of systems engineering. Implementations of the UCA have previously focused on using it to explore the process of self-replication in artificial life [19], [30], and in these systems the artefact machine D was either removed or represented trivially. For Systems Engineering, this machine moves front-and-centre, because it is the main goal of the analysis. Whereas systems engineering results in a siloed approach to design and consequently imposes additional constraints on the solution space, by adopting the more holistic approach described herein the use of Von Neumann's UCA model offers the possibility of allowing the complexity of the solution to increase proportionally to the design challenge, and that the design intent can then be embedded via evolutionary processes.

Acknowledgment

The authors would like to thank Peter Kilpatrick for his thoughts and comments on a early draft of this contribution. This work was funded by EPSRC, UK, grant reference EP/V007335/1 "RIED: Re-Imagining Engineering Design"

References

[1] S. J. Kapurch, *NASA Systems Engineering Handbook*. DIANE Publishing, Nov. 2010. [Online]. Available: https://play.google.com/store/books/details?id=2CDrawe5AvEC

[2] T. Weilkiens, J. G. Lamm, S. Roth, and M. Walker, *Model Based Systems Architecture, 2nd Edition*. Wiley, Apr. 2022.

[3] P. Kroes and A. Meijers, "The dual nature of technical artefacts," *Stud. Hist. Philos. Sci. B Stud. Hist. Philos. Modern Phys.*, vol. 37, no. 1, pp. 1–4, Mar. 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0039368105001032

[4] M. A. Price, T. T. Robinson, D. Soban, A. Murphy, C. G. Armstrong, R. McConnell, and R. Roy, "Maintaining design intent for aircraft manufacture," *CIRP Ann.*, vol. 62, no. 1, pp. 99–102, Jan. 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000785061300125X

[5] K. Fu, D. Moreno, M. Yang, and K. L. Wood, "Bio-Inspired design: An overview investigating open questions from the broader field of Design-by-Analogy," *J. Mech. Des.*, vol. 136, no. 11, p. 111102, Oct. 2014. [Online]. Available: https://asmedigitalcollection.asme.org/mechanicaldesign/article-abstract/136/11/111102/375236

[6] A. E. Eiben, S. Kernbach, and E. Haasdijk, "Embodied artificial evolution: Artificial evolutionary systems in the 21st century," *Evol. Intell.*, vol. 5, no. 4, pp. 261–272, Dec. 2012. [Online]. Available: http://dx.doi.org/10.1007/s12065-012-0071-x

[7] J. Cong, C.-H. Chen, X. Meng, Z. Xiang, and L. Dong, "Conceptual design of a user-centric smart product-service system using self-organizing map," *Advanced Engineering Informatics*, vol. 55, p. 101857, Jan. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034622003159

[8] Y. Li, L. Li, Q. Lin, K.-C. Wong, Z. Ming, and C. A. Coello Coello, "A self-organizing weighted optimization based framework for large-scale multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 72, p. 101084, Jul. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210650222000542

[9] J. M. Smith and E. Szathmary, *The Major Transitions in Evolution*. OUP Oxford, Oct. 1997. [Online]. Available: https://play.google.com/store/books/details?id=wP9QEAAAQBAJ

[10] T. Taylor, "Evolutionary innovations and where to find them: Routes to Open-Ended evolution in natural and artificial systems," *Artif. Life*, vol. 25, no. 2, pp. 207–224, 2019.

[11] H. H. Pattee, "Cell psychology: An evolutionary approach to the Symbol-Matter problem," in *LAWS, LANGUAGE and LIFE: Howard Pattee's classic papers on the physics of symbols with contemporary commentary*, H. H. Pattee and J. Raczaszek-Leonardi, Eds. Dordrecht: Springer Netherlands, 2012, pp. 165–179.

[12] X. Fan, W. Sayers, S. Zhang, Z. Han, L. Ren, and H. Chizari, "Review and classification of bio-inspired algorithms and their applications," *J. Bionic Eng.*, vol. 17, no. 3, pp. 611–631, May 2020. [Online]. Available: https://doi.org/10.1007/s42235-020-0049-9

[13] M. Price, W. Zhang, I. Friel, T. Robinson, R. McConnell, D. Nolan, P. Kilpatrick, S. Barbhuiya, and S. Kyle, "Generative design for additive manufacturing using a biological development analogy," *Finite Elem. Anal. Des.*, vol. 9, no. 2, pp. 463–479, Mar. 2022. [Online]. Available: https://academic.oup.com/jcde/article-abstract/9/2/463/6547705

[14] A. Lomas, "Morphogenetic vase forms," *Artificial Life Conference Proceedings*, vol. 31, pp. 523–530, Jul. 2019. [Online]. Available: https://www.mitpressjournals.org/doi/abs/10.1162/isal_a_00215

[15] A. McKay, S. Chase, K. Shea, and H. H. Chau, "Spatial grammar implementation: From theory to useable software," *Artif. Intell. Eng. Des. Anal. Manuf.*, vol. 26, no. 2, pp. 143–159, May 2012. [Online]. Available: http://dx.doi.org/10.1017/S0890060412000042

[16] J. Von Neumann, "Theory of Self-Reproducing automata," *cba.mit.edu*, 1966. [Online]. Available: http://cba.mit.edu/events/03.11.ASE/docs/VonNeumann.pdf

[17] W. R. Buckley, "On the complete specification of a von neumann 29-state selfreplicating cellular automaton," *Private communication, wrb@wrbuckley. com*, 2004.

[18] D. Baugh and B. McMullin, "The emergence of pathological constructors when implementing the von neumann architecture for self-reproduction in tierra," in *From Animals to Animats 12*. Springer Berlin Heidelberg, 2012, pp. 240–248. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33093-3_24

[19] E. B. Clark, S. J. Hickinbotham, and S. Stepney, "Semantic closure demonstrated by the evolution of a universal constructor architecture in an artificial chemistry," *J. R. Soc. Interface*, vol. 14, no. 130, May 2017. [Online]. Available: http://dx.doi.org/10.1098/rsif.2016.1033

[20] G. Hoyland, *Merlin: The Power Behind the Spitfire, Mosquito and Lancaster: The Story of the Engine That Won the Battle of Britain and WWII*. HarperCollins UK, 2020.

[21] A. Hodgkinson and A. Eyre-Walker, "Variation in the mutation rate across mammalian genomes," *Nat. Rev. Genet.*, vol. 12, no. 11, pp. 756–766, Oct. 2011. [Online]. Available: http://dx.doi.org/10.1038/nrg3098

[22] T. Robinson, I. Friel, C. G. Armstrong, A. Murphy, J. Butterfield, M. Price, and A. Marzano, "Computer-aided design model parameterisation to derive knowledge useful for manufacturing design decisions," *Proc. Inst. Mech. Eng. Pt. B: J. Eng. Manuf.*, vol. 232, no. 4, pp. 621–628, Mar. 2018. [Online]. Available: https://doi.org/10.1177/0954405417708218

[23] K. Deb and S. Gulati, "Design of truss-structures for minimum weight using genetic algorithms," *Finite Elem. Anal. Des.*, vol. 37, no. 5, pp. 447–465, May 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168874X00000573

[24] S. Cussat-Blanc, K. Harrington, and W. Banzhaf, "Artificial gene regulatory Networks-A review," *Artif. Life*, vol. 24, no. 4, pp. 296–328, 2018. [Online]. Available: http://dx.doi.org/10.1162/artl_a_00267

[25] S. Hickinbotham, R. Dubey, I. Friel, A. Colligan, M. Price, and A. Tyrrell, "Evolving design modifiers," in *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*. ieeexplore.ieee.org, Dec. 2022, pp. 1052–1058. [Online]. Available: http://dx.doi.org/10.1109/SSCI51031.2022.10022087

[26] L. Michaut, H. J. Jansen, N. Bardine, A. J. Durston, and W. J. Gehring, "Analyzing the function of a hox gene: an evolutionary approach," *Dev. Growth Differ.*, vol. 53, no. 9, pp. 982–993, Dec. 2011. [Online]. Available: http://dx.doi.org/10.1111/j.1440-169X.2011.01307.x

[27] O. Teboul, "Shape grammar parsing: Application to image-based modeling," Ph.D. dissertation, Ecole Centrale de Paris, Jun. 2002. [Online]. Available: https://theses.hal.science/tel-00628906/file/thesis_teboul.pdf

[28] S. Stepney and S. Hickinbotham, "Bio-Reflective architectures for evolutionary innovation," *The 2019 Conference on Artificial Life*, vol. 28, pp. 192–199, Sep. 2016. [Online]. Available: https://www.mitpressjournals.org/doi/abs/10.1162/978-0-262-33936-0-ch038

[29] A. E. Eiben and J. Smith, "From evolutionary computation to the evolution of things," *Nature*, vol. 521, no. 7553, pp. 476–482, May 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14544

[30] D. Baugh and B. McMullin, "Evolution of GP mapping in a von neumann self-reproducer within tierra," in *Artificial Life Conference Proceedings*. direct.mit.edu, 2013, pp. 210–217. [Online]. Available: https://direct.mit.edu/isal/proceedings-pdf/ecal2013/25/210/1901259/978-0-262-31709-2-ch032.pdf