A Scalable Solution to N-bit Parity via Artificial Development

Tüze Kuyucu, Martin A. Trefzer, Julian F. Miller and Andy M. Tyrrell

***Abstract*— The design of electronic circuits with model-free heuristics like evolutionary algorithms is an attractive concept and field of research. Although successful to a point, evolution of circuits that are bigger than a 3-bit multiplier is hindered by the scalability problem. Modelling the biological development as an artificial genotype-phenotype mapping mechanism has been shown to improve scalability on some simple circuit problems and pattern formations. As a candidate solution to the scalability issue, an artificial developmental system is presented.**

The presented artificial developmental system is shown to develop a scalable parity circuit, which could be infinitely developed to build a growing parity circuit, hence, represents a general, scalable solution to n-bit parity.

The result obtained further supports the artificial developmental system as a good candidate solution to the scalability problem in evolvable hardware.

I. Introduction

Since the invention of integrated circuits, the complexity of the circuits designed and manufactured have been exponentially increasing over time. Unfortunately, Evolutionary Computation (EC) has not been able to demonstrate a similar trend since the first use of Evolutionary Algorithms (EAs) as circuit design tools. The field of Evolvable HardWare (EHW), which encompasses the design of electronic circuits using EAs, has not yet achieved the design of complex circuits. Hence, enhancing the power of EAs in order to achieve more complex circuit designs has been a subject to extensive research in EHW [7], [8], [14], [17]–[19]. The ability to scale to larger and higher complexity circuits in a reasonable amount of time is termed scalability.

Obtaining scalable systems and pushing the complexity barrier further in the evolution of hardware systems is a major topic in EHW. In traditional digital circuit design, the design of complex circuits is done by partitioning the circuit into smaller modules that are more easily solvable. Some researchers investigated and introduced explicit modularity mechanisms for EC applications, which were shown to speed up evolution of the solutions to complex problems [8], [19]. However, even with the use of mechanisms that can create modules during evolution, the scalability problem is still not resolved.

It is argued by many that the lack of scalability in EHW is caused by the use of a direct genotype-phenotype mapping, and using biologically inspired techniques to aid the scalability of EHW systems is becoming more popular [1], [4]–[6], [11], [15], [20]. Among the various approaches, Artificial Developmental Systems (ADSs) stand out as promising genotype-phenotype mapping techniques. Using developmental mechanisms to map a relatively small genotype to large, complex phenotype is inspired by the way nature works to obtain large organisms.

As a promising genotype-phenotype mapping technique, which also inherently demonstrates modularity, development has the potential to help EAs in the design of higher complexity problems.

In this paper, we introduce a new ADS that models the biological developmental system including the Gene Regulatory Networks (GRNs). The paper will present a short review of the related literature in Section II followed by Section III, which will introduce the proposed artificial developmental system. In Section IV, it will be shown that the presented artificial developmental system is capable of developing scalable organisms that exhibit modularity. Finally further discussion of the results and related work, followed by conclusions and future work will be presented in Section V.

II. Development in Evolvable Hardware

The most common use of evolution in the design of electronic systems is in a *single-cellular* fashion; there is a direct genotype-phenotype relationship, which is equivalent to direct evolution with no developmental approach.

Such an approach requires the single cell to be highly complex, and creates an exponentially growing search space for evolution as the target system becomes more complex. Using nature as the inspiration and moving towards a multi-cellular system, the evolution of higher complexity electronic systems may become achievable. Realizing this some researchers already studied whether implementing an artificial developmental mechanism would improve scalability in EHW. A number of the published developmental models turned out to be quite promising [6], [11], [12].

In nature, biological organisms use interactive networks of genes, GRNs, in order to map a small genotype (DNA) to a complex phenotype [2]. The complex genotype-phenotype mapping in development allows the representation of a complex phenotype via a smaller genotype. The genotype in a developmental system does not necessarily grow with the size of the phenotype, thus scales better on complex problems [4]–[6], [9]. Development is also known to exhibit a modular behaviour while building a phenotype [9]. Modularity is a desired property of a scalable mechanism, and as it was mentioned earlier, it has previously been successfully introduced to EC [8], [19].

There are various models of artificial development in EHW, which range from systematic mechanisms that provide specific instructions that unfold the genotype in order to obtain the phenotype, to systems that closely model biological development.

All authors are with The Department of Electronics
University of York, UK
{tk519,mt540,jfm7,amt}@ohm.york.ac.uk.
This work is part of a project that is funded by EPSRC - EP/E028381/1.

Modelling biology more closely in electronic system design could be advantageous, due to the fact that biological development is a well established system that exhibits a high level of evolvability [3]. However, mimicking biology closely in the implementation of a developmental system can over-complicate the developmental system and render it practically unusable for Evolutionary Computation (EC) applications. Thus it may be more practical to have a simple developmental mechanism that is rather engineered for EC than copied from biology [15], [16].

III. The Artificial Developmental System

The developmental system presented here models the biological cell interactions and gene regulation closely in order to benefit from the complex interactions present in a developmental system, but it is kept simple enough that the overhead of the developmental system is not too high.

A. Cell Interactions and Gene Regulation

The artificial developmental system involves interactions at two different levels:

1) Interactions within the Cell: The state, and the structure of a cell is determined via the regulatory network present within that cell. Various chemicals interact with each other activating and inhibiting genes (the GRN rules), which are provided by the cell genome. The aforementioned chemicals are further detailed in Subsection III-B. An example GRN within a single cell is illustrated in Figure 1.

2) Interactions in the Organism: The interactions within the organism are the interactions amongst the cells themselves, and the environment. The communication mechanisms present amongst the cells are:

- **Diffusion:** Diffusion is achieved via the outward dispersion of the proteins present in cells; this provides a means of long distance communication amongst the cells. The diffusion process is carried out for all the proteins available in every cell, and it has been implemented in a simplistic way; where half of an available protein diffuses out equally to the four nearest neighbours of the source cell i.e. each neighbour obtains $\frac{1}{8}$ of the cell's protein. Protein levels are adjusted after the GRN has processed the new developmental step (including the diffusion stage) for all cells. Thus, the GRN always works with the original protein levels and the order of cell update should not bias the course of development.
- **Direct link communication:** This is achieved by establishing direct links between two neighbouring cells. A link between two cells is established as a result of a GRN activity in both of the cells that enable direct linking with their respective neighbours. Such a link provides a local interaction amongst the cells.

Using both local and long distance cell signalling mechanisms should provide symmetry breaking properties, allowing the ADS to generate irregular as well as regular cell patterns.

Each gene (rule) in the GRN system described here is represented by a binary string. Similar to biology the gene string is formed of two parts: precondition and postcondition. The postconditional part of the gene specifies the resulting actions of the gene in case it is activated. The pre-conditional part of the gene specifies the conditions that need to be met in order to activate the gene.

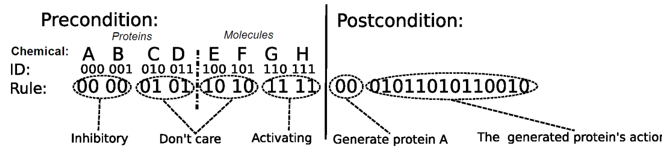

Fig. 2. An example gene formed of 32 bits is shown. The first 16 bits are reserved for the pre-conditional part, which specifies the rules to activate the gene. There are 8 chemicals (4 *proteins*, 4 *molecules*) defined in this figure. Each chemical's required presence or absence is specified by a 2 bit number, which provide two don't care states. In the event of a don't care state, the presence or absence of a chemical has no effect on the activation of the particular gene. The second 16 bits of the gene is reserved for the postconditional part, which provides the ID of the protein produced as a 2 bit number–this means that only the first four chemicals can be produced, which is then followed by a 14 bit number. The last 14 bits in the gene define the action of the protein produced.

Similar to the ADS used by Gordon [6], each gene is modeled as a rule that can be activated, inhibited or not affected by the known chemicals. For a gene to be activated all the activator chemicals must be present and all the inhibitor chemicals must be absent. Each chemical has a concentration, and it is considered to be present if its concentration is at or above the threshold level, otherwise it is considered absent. The postcondition of a gene defines the protein that is produced, and depending on the type of protein produced, the type of action it is going to take is also defined within the postcondition. Encoding of an example gene is illustrated in Figure 2.

B. Chemicals

The GRN system is composed of various proteins that are used to create rules and determine cellular actions. The proteins structure the cells, create a regulatory network, and are used by the cells to communicate with other cells. The proposed GRN system has four types of proteins:

1) **Plasmodesma Protein -** Plasmodesma proteins are inspired by the *Plasmodesmata* in plants - threads of cytoplasm that breach the cell wall and connect neighbouring cells (similar to gap junction in animal development) [10]. When a plasmodesma protein is produced, it forms a tunnel in one of the four cardinal directions, and if the neighbouring cell also has formed a tunnel in the corresponding direction, the two tunnels join together allowing the free passage of proteins between the two cells. After the two cells are connected by plasmodesmata (tunnels), they share their proteins. If the neighbouring cell space is not occupied, a cell division is initiated. During the cell division plasmodesmata is created connecting the two daughter cells (one of which now fills the empty cell space).
2) **Structuring Protein -** Structuring proteins are the

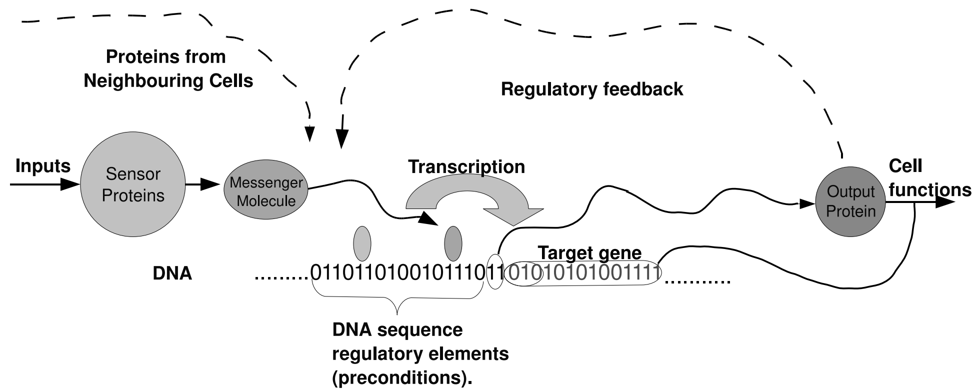

Fig. 1. The genes in a DeoxyriboNucleic Acid (DNA) sequence are activated via the correct protein activity in the GRN that cause transcription of the particular gene. In the example shown in the figure, the resulting activity of a GRN provides the correct transcription factors (the promoter proteins) to be bound at the start of a gene sequence. This then initiates the production of another protein, which can then affect the cell functionality using further information that is provided in the gene. The produced protein also creates a feedback to the GRN which regulates the transcription of further genes.

type of proteins that change the physical structure of the cell, i.e. the circuit. When a structuring protein is produced, it uses the further information provided by the gene to alter the circuitry within the cell.

3) **Sensor Protein -** The sensor proteins are produced by the GRN to act as sensors around the cell, which monitor outside activity. The sensor proteins produce different kinds of messenger molecules for different types of outside activity. This enables environmental factors to affect the GRN activity, creating a more interactive developmental system.
4) **Regulatory Protein -** Although every protein is a regulatory protein, there are proteins that act only as regulatory proteins. These proteins, just like every other protein, control the GRN activity by their presence or absence, but they do not have any other purpose.

Apart from the four proteins introduced, there are four other types of chemicals that behave similar to the proteins, however have different regulatory properties. Hence, these chemicals are referred to as "molecules". The said chemicals are called **Messenger Molecules:** The messenger molecules are produced by the sensor proteins that are affected by the outside activity. They can not be produced directly as a result of gene activity since their purpose is to regulate the gene activity via the environmental response, but they can still bind to activate or deactivate genes.

IV. Circuit Development

The experiments presented in this paper are initial experiments using the proposed developmental system, and they are carried out as software simulations. The developmental system is evolved to develop a phenotype in the form of a Cartesian Genetic Programming (CGP) netlist, which is a convenient way of simulating digital circuits [13].

For the experiments in this paper: the concentration of each protein is represented by an 8-bit unsigned integer; thus the maximum concentration is 255. The organism is initialised with one cell alive in the middle of the virtual cellular space.

TABLE I
The multiplexers used for the experiments presented.

MUX1	$(A \ \& \ \overline{C}) \| (B \ \& \ C)$
MUX2	$(A \ \& \ \overline{C}) \| (\overline{B} \ \& \ C)$
MUX3	$(\overline{A} \ \& \ \overline{C}) \| (B \ \& \ C)$
MUX4	$(\overline{A} \ \& \ \overline{C}) \| (\overline{B} \ \& \ C)$

The components used for building the structural part of the cells are 4 different types of MUXes, shown in Table I. The maximum organism size is predefined, and the cells are arranged in columns to form an organism of size $n \times n$.

A. Developing a Parity Solving Organism

In order to demonstrate the scalability of the GRN mechanism, a 2 cell organism is evolved that develops to function as an XOR gate. The organism is formed of 5 genes using 4 proteins (1 plasmodesma, 1 structuring, 2 regulatory), and it develops to an adult organism in 3 developmental steps. It was evolved with the maximum number of cells limited to 4. The evolved organism is discovered to have the potential to grow to become an n-bit parity solving organism if the maximum number of cells available were to be increased. As shown in Figure 3, the GRN of the evolved organism keeps replicating an XOR gate for each row of cells.

The growth of the XOR organism in more detail is shown in Figure 3. The example shown in Figure 3 demonstrates the proposed developmental system's ability to achieve modularity.

In the example shown in Figure 3, at developmental step 1 only one cell is alive in the organism. Looking at the GRN rules it can be seen that, Gene 0 and Gene 3 do not need any promoters in order to activate and the suppressor proteins initially do not exist (initially none of the proteins exist in the system). Thus Gene 0 and 3 would be active at the first stages of the GRN, while the others are inactive. Gene 3 would produce Plasmodesma Protein targeted south (this information is encoded in the postconditional bitstring, which is not shown here), which would then trigger a growth

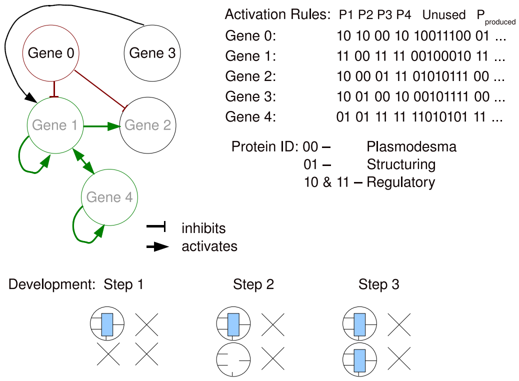

Fig. 3. An example 2 cell organism that was evolved to act as an XOR gate is shown, with the list of its evolved genes, and the GRN graph for the alive cells (identical for both cells in this case).

process. On developmental step 2 the second cell grows and shares its proteins with the parent cell. By step 3 the organism stabilizes with two identical cells, each implementing an XOR with a MUX but only the daughter cell being connected to the output. The developed organism in this example demonstrates a modular and redundant behaviour by replicating the same physical structure in both of the cells, which are desirable features for a scalable system. More importantly, if the organism was allowed to develop further, the daughter cell from step 2 would trigger a growth process into its southern neighbour, which would then become a chain of infinite growth events, resulting in the growth of an infinitely sized parity solving organism. Hence achieving an n-bit Parity circuit in N developmental steps.

V. Discussion and Future Work

A new artificial developmental system has been introduced, and it was shown that the proposed developmental system is scalable. The mechanisms of the presented developmental system are designed to be biologically defensible, but they have been kept simple enough to keep them computationally inexpensive and to make them suitable for hardware implementation.

To the authors' knowledge, this is the first developmental system that was shown to be capable of developing to n-bit parity circuit without re-evolving for different n values.

Although solving n-bit parity itself is a trivial problem, demonstrating that the presented developmental system possesses the potential to scale on circuit problems is a notable achievement for the initial stages of an ADS aimed for circuit development.

Future work will include investigating suitable applications for development. Also, the application of the ADS on hardware will be tested; it is expected to be faster and capable of running online (no predetermined developmental steps) when run on hardware. It will then be further investigated for fault tolerance, scalability and adaptivity, and applied to real world applications.

References

[1] P. Bentley and S. Kumar, "Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem," in *Proceedings of the Genetic and Evolutionary Computation Conference*, vol. 1. Orlando, Florida, USA: Morgan Kaufmann, 13-17 1999, pp. 35–43. [Online]. Available: citeseer.ist.psu.edu/bentley99three.html

[2] E. Davidson, *The Regulatory Genome: Gene Regulatory Networks In Development And Evolution*, 1st ed. ACADEMIC PRESS, 2006.

[3] R. Dawkins, *On Growth, Form and Computers*. Elsevier Academic Press, 2003, ch. The evolution of Evolvability, pp. 239–255.

[4] F. Dellaert and R. Beer, *Toward an Evolvable Model of Development for Autonomous Agent Synthesis*. MIT Press Cambridge, 1994. [Online]. Available: citeseer.ist.psu.edu/dellaert94toward.html

[5] P. Eggenberger, "Evolving morphologies of simulated 3d organisms based on differential gene expression," in *Proceedings of 4th European Conference on Artificial Life*, 1997, pp. 205–213.

[6] T. G. W. Gordon, "Exploiting development to enhance the scalability of hardware evolution," Ph.D. dissertation, University College London, July 2005.

[7] P. C. Haddow and G. Tufte, "Bridging the genotype-phenotype mapping for digital fpgas," in *EH '01: Proceedings of the The 3rd NASA/DoD Workshop on Evolvable Hardware*. Washington, DC, USA: IEEE Computer Society, 2001, p. 109.

[8] J. R. Koza, *Genetic programming II: automatic discovery of reusable programs*. Cambridge, MA, USA: MIT Press, 1994.

[9] S. Kumar and P. Bentley, Eds., *On Growth, Form and Computers*. Elsevier Academic Press, 2003.

[10] O. Leyser and S. Day, *Mechanisms in Plant Development*. Blackwell, 2003.

[11] H. Liu, "Biological development model for the design of robust digital system," Ph.D. dissertation, University of York, September 2007.

[12] J. F. Miller, "Evolving developmental programs for adaptation, morphogenesis, and self-repair," in *7th European Conference on Artificial Life*. Springer LNAI, 2003, pp. 256–265.

[13] J. F. Miller and P. Thomson, "Cartesian genetic programming," in *Genetic Programming, Proceedings of EuroGP'2000*. Springer-Verlag, 2000, pp. 121–132.

[14] M. Murakawa, S. Yoshizawa, I. Kajitani, T. Furuya, M. Iwata, and T. Higuchi, "Hardware evolution at function level," in *PPSN IV: Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*. London, UK: Springer-Verlag, 1996, pp. 62–71.

[15] D. Roggen, "Multi-cellular reconfigurable circuits: Evolution morphogenesis and learning," Ph.D. dissertation, EPFL, 2005. [Online]. Available: citeseer.ist.psu.edu/roggen05multicellular.html

[16] G. Tempesti, D. Mange, E. Petraglio, A. Stauffer, and Y. Thoma, "Developmental processes in silicon: An engineering perspective," in *EH '03: Proceedings of the 2003 NASA/DoD Conference on Evolvable Hardware*. Washington, DC, USA: IEEE Computer Society, 2003, p. 265.

[17] J. Torresen, "A divide-and-conquer approach to evolvable hardware," in *ICES '98: Proceedings of the Second International Conference on Evolvable Systems*. London, UK: Springer-Verlag, 1998, pp. 57–65.

[18] V. K. Vassilev and J. F. Miller, "Scalability problems of digital circuit evolution: Evolvability and efficient designs," in *EH '00: Proceedings of the 2nd NASA/DoD workshop on Evolvable Hardware*. Washington, DC, USA: IEEE Computer Society, 2000, p. 55.

[19] J. Walker and J. Miller, "Evolution and acquisition of modules in cartesian genetic programming," in *EuroGp*, vol. 3003/2004. Springer Berlin / Heidelberg, 2004, pp. 187–197.

[20] S. Zhan, J. F. Miller, and A. M. Tyrrell, "A developmental gene regulation network for constructing electronic circuits," in *8th International Conference on Evolvable Systems: From Biology to Hardware*. Springer-Verlag, 2008, pp. 177–188.