Towards In Vivo Genetic Programming: Evolving Boolean Networks to Determine Cell States

Towards In Vivo Genetic Programming: Evolving Boolean Networks to Determine Cell States

Nadia S. Taou and Michael A. Lones

School of Mathematical and Computer Sciences
Heriot-Watt University, Edinburgh, EH14 4AS, UK
{nt2, m.lones}@hw.ac.uk

Abstract. Within the genetic programming community, there has been growing interest in the use of computational representations motivated by gene regulatory networks (GRNs). It is thought that these representations capture useful biological properties, such as evolvability and robustness, and thereby support the evolution of complex computational behaviours. However, computational evolution of GRNs also opens up opportunities to go in the opposite direction: designing programs that could one day be implemented in biological cells. In this paper, we explore the ability of evolutionary algorithms to design Boolean networks, abstract models of GRNs suitable for refining into synthetic biology implementations, and show how they can be used to control cell states within a range of executable models of biological systems.

Keywords: Gene regulatory networks, Boolean networks, control, evolutionary algorithms

1 Introduction

Gene regulatory networks (GRNs) are biochemical systems that process information and generate complex responses within biological organisms. In effect, they are the "genetic programs" of biological cells. Because of this, GRNs have long been a source of inspiration to the genetic programming community, with the first papers using representations motivated by GRNs published around the turn of the millenium [24,3]. Recently this interest has started to blossom, with a number of different research groups looking at how GRN-based approaches can be used to solve computational problems within computers [18]. The motivations for this are various: to increase the evolvability of genetic programming [3], to increase the robustness of executional systems [27], to support the evolution of complex computational behaviours [19], and to increase the compactness or efficiency of computation [32]. However, regardless of the motivation, the focus of this research has been pretty singular: evolving GRN models that will be executed *in silico*.

In the last decade, there has been considerable progress in the field of synthetic biology [17]. An important activity within synthetic biology is the design and assembly of novel biochemical pathways that can be deployed within an

existing cell in order to change its behaviour. Typically this is aimed at implementing medical interventions in a way that is more precise and/or effective than conventional therapeutic methods. At the moment, these synthetic circuits usually take the form of conventional digital designs (i.e. feed-forward logic circuits), which can be coupled to the existing gene regulatory pathways through the control of particular transcription factors. These logic circuits are themselves implemented using proteins and nucleic acids, and can be deployed into cells using various mechanisms, including customised viruses [17,28]. It will likely soon reach the point where a synthetic circuit can be delivered to a particular cell within the human body.

Whilst the GP community is becoming more interested in computational representations found in biological cells, the synthetic biology community mainly focuses on using computational representations found in silicon systems. Given that we have found computational models of GRNs to exhibit desirable properties (e.g. compact expressiveness, intrinsic fault tolerance) that are not necessarily found in more conventional models of computation, this might seem like an odd situation. In particular, why not use the existing design principles of biological cells and build synthetic GRNs rather than Boolean logic circuits? One answer to this question is that most synthetic biologists (and indeed most people) do not understand the design principles of GRNs, and hence can not design synthetic GRNs that have particular computational behaviours. This would appear to open up an opportunity for a community that can design GRNs that have particular computational behaviours, and this is almost exactly what we have been doing in recent years in the GP community, albeit with a fairly ad hoc group of GRN models and target behaviours. In essence, we suggest there is significant scope for using GP (and related approaches) to evolve programs that could be run *in vivo* — that is, within biological cells — and hence close the circle from biological inspiration to computational optimisation and back again to biology.

In this paper, we present work on using evolutionary algorithms (EAs) to design Boolean networks (BNs) that have a particular biological function. BNs are a class of abstract GRN models that are known to have steady-state equivalence to more detailed quantitative models [35], such as systems of differential equations, and have been successfully used to model various biological networks [26]. Since they are composed of Boolean functions, they also lend themselves well to implementation using existing synthetic biology techniques [28], so in principle evolved models could be refined into biological implementations. This is not necessarily the case with the more complex models currently used in the GP community, and hence why we focus on this relatively abstract class of GRN models.

An important problem in biology is the control of cell state [4]. This plays a role in many diseases: for instance, many cancers are thought to be caused by a cell transitioning to an abnormal cell state [13], and a potential cure would involve guiding its transition back to a normal cell state. Transitions in cell state are governed by a cell's GRN, and it is necessary to intervene in the GRN's natu-

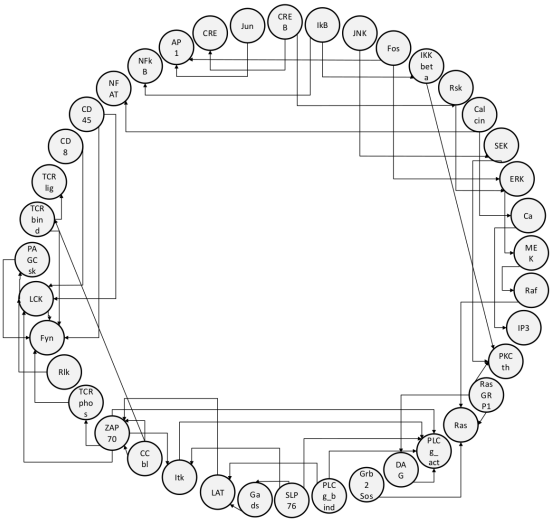
ral dynamics in order to change this. At present, such interventions are typically achieved by using a drug to target a single gene, forcing it to be permanently on or off. However, this is a rather blunt form of intervention and unlikely to be sufficient for causing large-scale changes within a cell's behaviour. More effective forms of intervention would involve the coordinated targeting of multiple genes in a particular temporal pattern. In effect, this requires a control strategy, and consequently much of the work in this area has focused on using conventional control techniques to generate appropriate patterns of intervention [22,8]. However, this is an NP hard problem [1], meaning that in practice exact analytical solutions can only be found for small systems. This in itself suggests a potential role for metaheuristics such as EAs, which are often used to address problems where analytical solutions are infeasible.

In previous work [29,30], we have shown that EAs can be used to design BNs that can carry out control (i.e. generate a series of control interventions) when coupled to a target BN. The target Boolean networks, in this case, were randomly sampled from the space of all BNs of a given size, giving an estimate of the general ability of evolved BNs to influence the dynamics of other BNs. In the work reported in this paper, by comparison, we focus on the control of actual models of biological regulatory networks, and show that evolved BNs are able to govern these systems so that they transition to a specific attractor corresponding to a particular cell state. In this sense, they are much closer to being viable "genetic programs".

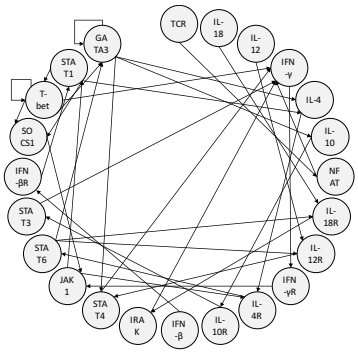
Section 2 presents a brief introduction to BNs and describes the BN models of biological networks that serve as control targets in this work. Section 3 describes the experimental methodology. Sections 4 and 5 present results and discussion, and Section 6 concludes.

2 Boolean Networks

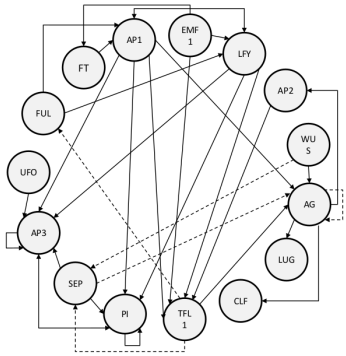
A Boolean network (BN) is a discrete-time non-linear dynamical system represented as a directed graph $G(V, E)$ composed of nodes, or vertices, V and edges E [14,7]. The time evolution of a BN is expressed by a set of Boolean functions f_i, $i = 1, 2, 3, ...$ Each BN node has a binary state s which is updated synchronously according to its Boolean function and the states of the k input nodes that are connected to it. Formally, $s(t+1) = f_i(s(t))$, where s is a set of network states, N is the number of nodes, $s \in \{0,1\}^N$, $t = 0, 1, 2, 3, 4, ...$ is the discrete time, and $f_i : \{0,1\}^N \rightarrow \{0,1\}$. Since a BN is deterministic $s(t+1)$ is only determined by $s(t)$. The possible number of Boolean functions is 2^{2^k}, and the state space is finite and equal to 2^N. Since the state space is finite, states must eventually be repeated, leading to temporal structures called attractors. When used to model GRNs, these attractors can be interpreted as the stable states (or cell types) of a cell [12].



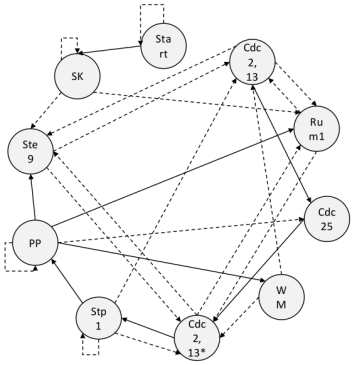
(a) T cell receptor signalling



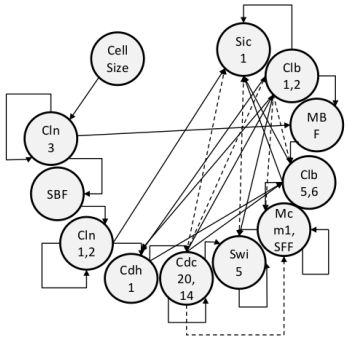
(b) T helper cell differentiation



(c) Arabidopsis thaliana



(d) Fission yeast cell cycle



(e) Budding yeast cell cycle

Fig. 1: Biological models used as case studies in this work.

2.1 Boolean Models of Biological Networks

To evaluate the ability of our control method in realistic biological situations, we selected five BNs from the literature that model well-known genetic regulatory systems [21,15,6,2]. Several factors motivated this choice. First, we wanted to look at the effect of network size, and the selected BNs vary in size from 10 nodes to 40 nodes. Second, it is important to show that the method works on systems with different state space structures. To address this, we chose BNs with different numbers of stable states, since this gives some indication of the complexity of the dynamics: the selected BNs have between 3 and 13 stable states, all of which are point attractors (i.e. a single repeating expression state). Finally, the chosen models are biologically diverse, capturing a range of different biological processes (morphogenesis, signalling and cell cycle regulation) that occur in a range of different species (single-celled organisms, plants, and animals). Each is briefly described in this section.

T cell receptor signalling pathway T cells are a subgroup of white blood cells that play a crucial role in the adaptive immune response, helping to protect the host against different pathogens such as virus and bacteria. The inappropriate activation of a T cell can lead to various autoimmune diseases. T cell receptor (TCR) is a membrane protein found on the surface of T cells which contributes to their activation by recognising antigen. A BN of the TCR signalling pathway is described in [15] and is depicted in Fig. 1a. It comprises 40 genes and has 8 point attractors, corresponding to different activation and proliferation cell states. See [15] for details of Boolean functions.

T helper cell differentiation network T helper cells, commonly called Th cells, are a type of T cell that plays a critical and key role in the adaptive immune system, where they help the immune activities of other immune cells such as B cell antibodies, plasma cells and cytotoxic T cells. T helper cells differentiate into one of the largest subcategories of cells, for example TFH, Th1, Th2, Th3, Th9 and Th17, which produce and release several types of T cells cytokines to regulate immune responses. A BN model of Th cell differentiation was developed in [21]. This model, depicted in Fig. 1b, captures the activities of 23 genes and has three point attractors, corresponding to different Th cell types. See [21] for details of Boolean functions.

Flower morphogenesis in arabidopsis thaliana Morphogenesis, the development of an organism's form through the process of cell differentiation, is an important component of multicellular organisms, and often plays a role in disease development. The most widely studied models of morphogenesis concern flower development in plants, and particularly within the model species *arabidopsis thaliana*, a small flowering plant. Flower morphogenesis occurs during the entire life cycle from groups of undifferentiated cells known as meristems. These develop into various different cell types in order to form the organs of a

flower, for example sepals, petals, stamens and carpels. A BN model of flower morphogenesis in arabidobis thaliana is described in [2]. It comprises 15 genes and has 10 point attractors, each corresponding to a different cell type. See Fig. 1c. Details of Boolean functions can be found in [20] and [2].

Fission yeast cell cycle regulation Fission yeast is the common name of *schizosaccharomyces pombe*, a unicellular eukaryote whose cells are rod-shaped and divide by medial fission. It is a well known system used to study cell growth and division, mainly because of their simple shape and their place within the eukaryotic lineage. The fission yeast cell cycle is the sequence of events that occur in a cell leading to duplication of all its components and its division into two almost identical daughter cells. A BN model of fission yeast cell cycle regulation is given in [6]. It is formed by 10 genes and has 13 point attractors, corresponding to different stable cell states within the cell cycle. See Fig. 1d. Details of Boolean functions can be found in [6].

Budding yeast cell cycle regulation Budding yeast is another species of yeast that has been widely used to study the eukaryotic cell life cycle. As the name implies, new cells form as a bud that grows from an existing cell, rather than undergoing fission. A BN model of budding yeast cell cycle regulation is described in [6]. It has 12 genes and 7 point attractors. See Fig. 1e. Details of Boolean functions can be found in [16].

3 Evolutionary Methods

In this paper we optimise Boolean networks to control Boolean models of real biological networks. We focus on applying a control intervention (i.e. a series of perturbations) that guides a trajectory of a controlled BN from a random initial state to a particular stable state (attractor) in its state space. This is done by coupling a controller BN to the controlled BN (see Fig. 2). During the course of its execution, the controller BN generates a series of interventions by setting the states of one or more target nodes (referred to as coupling terms) within the controlled BN.

The topology, node functions, coupling terms, and timing parameters of the controller BN are optimised using an EA. The effectiveness of a controller's interventions are measured using a fitness function that returns the Euclidean distance between the target state and the actual state that is reached by the end of a control period of 100 time steps of the controlled BN. This is linearly scaled to the interval $[0, 1]$, where a fitness of 1.0 indicates that the target state was reached. The fitness distribution over 20 runs is used to give an estimate of the ability of the EA to find a controller BN that can control a specified controlled BN so that it reaches a specified stable state. This is repeated for each stable state of each target network.

A controller BN has two evolved timing parameters, each within the range $[1, 50]$. The first timing parameter determines the number of time steps the

controller BN will perform for each time step of the controlled BN, i.e. the relative speed of the controller. The second timing parameter indicates how frequently the controller BN is executed, in terms of the number of time steps of the controlled network, i.e. how often it intervenes.

A controller RBN is represented as an array of nodes, each comprising a Boolean function number between 0 and 2^{2^k}, an initial state, and a set of input nodes, where each input is indicated by its position within the array. For these experiments, k is fixed at 2 (the edge-of-chaos regime, where computational behaviours are hypothesised to be maximal [11]) and the controller has a fixed length of 15 nodes. In previous work, we have found this length to offer a fair trade-off between expressiveness and search space size [29]. The solution chromosome also contains the timing parameters and the coupling terms. See Fig. 3. A generational evolutionary algorithm is executed for 100 generations each run, with a population size of 500, tournament selection ($n = 3$), uniform crossover ($p_c = 0.15$), point mutation ($p_m = 0.06$) and elitism ($n = 1$).

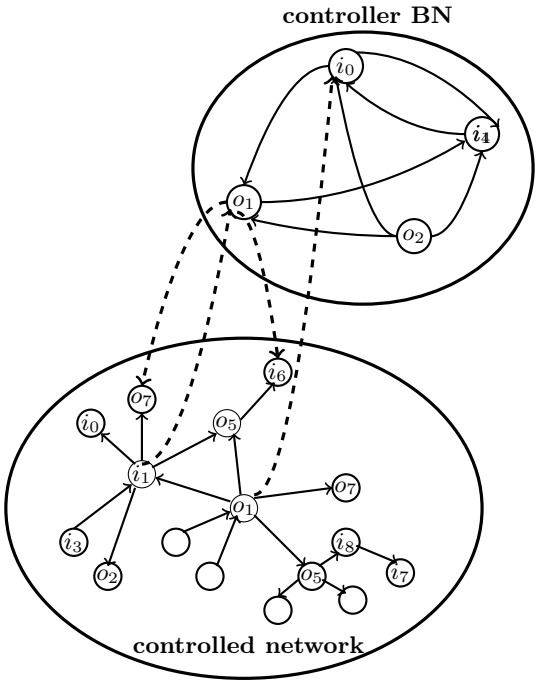


Fig. 2: Evolved controller Boolean network (representing a synthetic GRN) coupled to a controlled Boolean network (representing a native biological regulatory network). Coupling between the two networks is implemented by copying the expression states from designated nodes in the controller to designated nodes in the controlled network (depicted as dotted arrows in this diagram) at specified intervals.

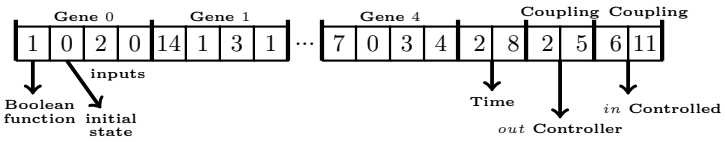


Fig. 3: Example of a Boolean network's genetic representation. Since $k = 2$, functions are numbered between 0 and 15. The timing and coupling terms indicate that this network is iterated twice each time it is executed, it is executed every 8 steps of the controlled network, its control outputs (interventions) are copied to nodes 2 and 5 of the controlled network, and its feedback (in) inputs from the controlled network are copied from nodes 6 and 11.

4 Results

Tables 1–5 present summary statistics for the fitness distributions of both the natural (i.e. how close it gets to the target state in the absence of control) and controlled dynamics of each case study BN for each target stable state. Table 6 summarises these results, showing the mean fitness achieved and the number of target states reached (with and without control) within each biological network. Without control, only a small number of these attractors were reached (4/32). Even when they were reached, the standard deviations in fitness (i.e. in distance from the target) were generally large. This indicates that, for a particular evolutionary run, most randomly sampled initial states will not be within the basin of attraction of the target attractor, and hence the control problems are non-trivial.

In all the case study BNs, the EA was able to find controllers that can target the majority of the steady states from a random initial state. Where the target was reached, the standard deviation between runs tended to be low, meaning that most runs are able to find BNs with optimal, or at least near-optimal, control strategies: the maximum likelihood estimation is 1.0 when BN controllers are successfully found and 0.9 otherwise. In all cases, the evolved controllers guided the system closer to the target states than could be achieved in the absence of control (see p-values in tables).

Some target BNs appear to be harder to control than others. The arabidopsis thaliana and T cell receptor signalling networks both have three steady states which were not reachable by the evolved controllers; although, in both cases, the systems could be controlled to states not far from the target state. However, there does not appear to be a simple relationship between the difficulty of the control task and the number of attractors: for example, the fission yeast BN, which has the most attractors, was the easiest to control. There is, however, a mild negative correlation (-0.23) between network size and control fitness, and indeed the largest network (T cell receptor signalling) was one of the hardest to control.

Although the majority of target states could be reached, the evolved controllers were not able to reach all target states. Further research is required to understand exactly why this is the case, though we can speculate it is likely due to at least two reasons. First, in some target networks, the majority of random states may fall far from the basin of attraction of a particular stable state, making the problem intrinsically hard when an arbitrary initial state is chosen. Second, transitions between states in BNs are typically not between adjacent states, meaning that in many cases there will not be valid transitions from states which differ by a single bit from the target: this is likely to lead to deceptive local optima in the state space. If this is the case, there may be scope for using diversity preservation techniques (e.g. crowding, fitness sharing) to navigate around local optima during optimisation. This is something we plan to look at in future work. Nevertheless, the results are promising, and demonstrate that even basic evolutionary algorithms can solve state space targeting problems, and can do so in a way that does not require *a priori* understanding of the structure of the state space.

Table 1: Fitness distributions for the T cell receptor signaling pathway control problem, indicating the ability of trajectories to reach each of the system's stable states both with and without control. A fitness of 1 is optimal.

	Control			No Control			
Attractors	Mean	Std. Dev	Max	Mean	Std. Dev	Max	p-value
1	0.996	0.009	1	0.851	0.060	0.950	1.278×10^{-08}
2	0.975	0.026	1	0.850	0.034	0.900	1.596×10^{-08}
3	0.996	0.009	1	0.843	0.075	0.975	2.745×10^{-08}
4	0.975	0	0.975	0.869	0.066	0.950	3.664×10^{-09}
5	0.996	0.009	1	0.861	0.066	0.950	9.115×10^{-09}
6	0.969	0.010	0.975	0.917	0.055	0.975	1.49×10^{-05}
7	0.975	0	0.975	0.868	0.048	0.950	5.66×10^{-09}
8	1	0	1	0.844	0.051	0.950	3.073×10^{-09}
General Mean	0.985	0.008	0.990	0.863	0.057	0.950	1.872×10^{-06}

Table 2: T-helper cell differentiation

	Control			No Control			
Attractors	Mean	Std. Dev	Max	Mean	Std. Dev	Max	p-value
1	0.972	0.065	1	0.553	0.067	0.652	1.094×10^{-08}
2	1	0	1	0.601	0.179	0.826	3.823×10^{-09}
3	0.867	0.0446	0.913	0.510	0.161	0.826	5.285×10^{-08}
General Mean	0.946	0.036	0.971	0.554	0.135	0.768	2.253×10^{-08}

Table 3: Arabidopsis thaliana flower morphogenesis

	Control			No Control			
Attractors	Mean	Std. Dev	Max	Mean	Std. Dev	Max	p-value
1	1	0	1	0.863	0.137	1	1.094×10^{-08}
2	0.926	0.030	0.933	0.561	0.124	0.800	2.75×10^{-09}
3	0.989	0.033	1	0.635	0.100	0.733	5.693×10^{-09}
4	0.933	0	0.933	0.800	0.049	0.866	2.726×10^{-09}
5	1	0	1	0.835	0.144	0.933	2.549×10^{-09}
6	0.933	0	0.933	0.217	0.150	0.800	3.027×10^{-09}
7	1	0	1	0.919	0.042	1	3.3×10^{-08}
8	1	0	1	0.624	0.074	0.733	3.062×10^{-09}
9	1	0	1	0.382	0.184	0.933	4.479×10^{-09}
10	0.996	0.015	1	0.256	0.059	0.333	4.45×10^{-09}
General Mean	0.977	0.0078	0.979	0.609	0.110	0.831	7.263×10^{-09}

Table 4: Fission yeast cell cycle

	Control			No Control			
Attractors	Mean	Std. Dev	Max	Mean	Std. Dev	Max	p-value
1	1	0	1	0.442	0.285	1	3.916×10^{-08}
2	1	0	1	0.321	0.171	0.900	2.25×10^{-09}
3	0.921	0.042	1	0.594	0.102	0.700	6.823×10^{-09}
4	0.994	0.022	1	0.447	0.219	0.900	4.107×10^{-09}
5	0.994	0.022	1	0.505	0.246	0.900	5.482×10^{-09}
6	1	0	1	0.573	0.133	0.900	1.921×10^{-09}
7	1	0	1	0.484	0.121	0.800	2.377×10^{-09}
8	0.900	0	0.900	0.763	0.095	0.900	2.088×10^{-09}
9	1	0	1	0.600	0.124	0.800	12.483×10^{-06}
10	0.984	0.0373	1	0.405	0.154	0.900	2.457×10^{-09}
11	0.921	0.041	1	0.552	0.134	0.800	1.274×10^{-08}
12	1	0	1	0.382	0.184	0.933	8.583×10^{-09}
13	0.994	0.022	1	0.536	0.134	0.800	3.873×10^{-09}
General Mean	0.997	0.014	0.992	0.508	0.161	0.864	1.980×10^{-07}

Table 5: Budding yeast cell cycle

	Control			No Control			
Attractors	Mean	Std. Dev	Max	Mean	Std. Dev	Max	p-value
1	1	0	1	0.543	0.165	0.666	2.788×10^{-09}
2	1	0	1	0.627	0.321	0.916	2.088×10^{-09}
3	1	0	1	0.442	0.416	0.916	2.25×10^{-09}
4	1	0	1	0.500	0.328	0.833	2.544×10^{-09}
5	1	0	1	0.605	0.393	0.916	2.859×10^{-09}
6	0.916	0	0.916	0.521	0.249	0.750	2.335×10^{-09}
7	1	0	1	0.434	0.479	1	1.036×10^{-05}
General Mean	0.988	0	0.988	0.524	0.335	0.855	1.482×10^{-06}

Table 6: Summary of the results, showing the mean fitness (1 is optimal) across all runs, and the number of attractors reached, for each case study BN both when under the control of an evolved BN and when following its natural dynamics (no control) from a random initial state.

		Mean Fitness		Attractors Reached		
Network name	Size	Control	No Control	Total	Control	No control
Fission yeast cell cycle	10	0.997	0.508	13	12	1
Budding yeast cell cycle	12	0.988	0.524	7	6	1
Arabidopsis thaliana	15	0.977	0.609	10	7	2
T helper cell differentiation	23	0.946	0.554	3	2	0
T cell receptor signalling	40	0.985	0.863	8	5	0

5 Discussion

The results of this study suggest that it is possible to evolve synthetic GRN models that have specific, biologically-relevant, behaviours. This is not the first time that EAs have been used to design and optimise GRNs for use within a synthetic biology context. For example, a number of different research groups have previously used EAs to design GRN models that have simple dynamical behaviours such as oscillation and bistability [9,23,10]. Nevertheless, unlike these earlier studies, the synthetic GRN models evolved in this work have behaviours that could reasonably be described as computational or programmatic, since controllers are essentially programs that carry out decisions based on their inputs.

This work is very much motivated by previous work in the GP community where GRN models have been used to carry out computation. Control, in particular, has been a recurring application in this nascent research field, with GRN models evolved to solve control tasks in robotics [32], computer gaming [27], and chaotic systems [19], to name but a few. This seems natural, since control is one of the principal behaviours carried out by biological GRNs. However, evolved GRNs have also been used to solve more diverse tasks (e.g. image compression [33]) and theoretical studies have shown that GRN models such as BNs are computationally universal [18], so in principle evolved synthetic GRNs could be used to carry out a much broader range of computational tasks, and perhaps even be used as the basis of general-purpose cellular computers. There is also no reason to limit the scope of this research to GRNs. BNs, for example, can be used to model other important biological networks, such as intracellular signalling networks.

In this work, we intentionally used a standard evolutionary algorithm and a linear solution representation (essentially a genetic algorithm) in order to keep things simple. In practice, there is plenty of scope for using more advanced approaches. Notably, there has been a lot of work on evolving network structures, and much of this would be directly applicable especially if we aim to evolve larger,

more complex networks. Recent work in applying NEAT to GRN models [5], and applying Cartesian GP to recurrent networks [34] seem particularly relevant.

This is still early work, and there remains significant work to be done to show that this is a viable approach to designing synthetic GRNs. Initially, we plan to study the evolved controllers in order to gain insight into the nature (and diversity) of the computational behaviours that are carried out when solving these control tasks. However, we also need to take into account biological constraints when evolving controllers: for instance, restricting coupling terms to biologically accessible targets (since currently any node in the target network can be used for coupling), and focusing on biologically-meaningful initial conditions rather than randomly sampling starting states.

Whilst this will help to build confidence that evolved controllers are doing something useful and viable, we also need to demonstrate that the evolved controllers are robust. There are several aspects to this. First, there is the generality of a controller's behaviour; for instance, can a single controller tolerate different initial conditions? Second, there are the differences between simulation and reality. Research in evolutionary robotics has shown that this kind of "reality gap" can restrict the generality of evolved controllers. We know from existing research [25] [27] that GRN models are less susceptible to this problem, given their natural robustness. However, we also need to consider that the simulation environment used in this work is quite different to biological reality. For instance, biological cells are stochastic environments, both in terms of what occurs and when things occur. We might address this, for example, by using probabilistic BN models. There is also the question of how much confidence we have in the executable model used to evaluate a BN model, since this will determine how much confidence we have in the evolved model. However, this is a more general issue in biological modelling, and there has been significant progress in developing reliable executable models of biological systems [31].

Refining evolved BN programs into actual synthetic biology realisations would involve a number of extra challenges. For instance, in this work we evolved timing parameters to allow the controller and controlled systems to operate over different timescales. This may also be possible to do within synthetic biology implementations, e.g. using RNA interference rather than transcription factors to speed up the controller's logic, but it would not be trivial. Another issue might be limitations placed on the controller's size or topology due to the difficulty of avoiding cross-talk within synthetic biology circuits.

6 Conclusions

The control of a cell's state is an important problem: it is instrumental for controlling many disease processes, yet in practice it is very difficult to find a series of interventions that will guide a cell between two different states. In this paper, we describe a novel approach to solving this problem which involves optimising a synthetic gene regulatory network which is then used to generate a pattern of interventions based on the state of a target cell. The approach is evaluated

using computational simulation, representing the gene regulatory network as a Boolean network, and the target cells as executable Boolean models. An evolutionary algorithm is then used to carry out optimisation of the Boolean network. In the majority of the case studies we looked at, the evolutionary algorithm was able to find Boolean networks that could successfully guide the target cell model from a randomly sampled initial state to a biologically-meaningful cell state.

The choice of Boolean networks is not arbitrary. The fact that they are constructed from Boolean logic gates means that there is a potential pathway from model to biological implementation through the use of existing synthetic biology principles. The choice of an evolutionary algorithm is also not arbitrary, and is motivated by a larger body of work in the field of genetic programming which is concerned with using evolutionary algorithms to design programmatic behaviours. In recent years, the genetic programming community has increasingly made use of models of gene regulatory networks to represent evolving computation. In addition to being intrinsically evolvable, these representations have also proved able at expressing complex computational behaviours that are robust yet compact. However, to our knowledge, this is the first time that evolutionary algorithms have been used to design actual "genetic programs".

From this perspective, it is interesting to note that synthetic biology focuses on implementing feed-forward logic circuits and traditional models of computation within biological cells, rather than using native biological design principles. This is in contrast to the opposing direction of travel in the genetic programming community. There are various reasons for this, but a significant factor is the difficulty of designing gene regulatory networks, which are based around principles of non-linear dynamical systems rather than well understood digital design principles. However, the ability of evolutionary algorithms to optimise these structures suggests that the genetic programming community could play an important role in designing programs that will one day run *in vivo* within biological cells.

References

1. Akutsu, T., Hayashida, M., Ching, W.K., Ng, M.K.: Control of boolean networks: hardness results and algorithms for tree structured networks. Journal of Theoretical Biology 244(4), 670–679 (2007)
2. Alvarez-Buylla, E.R., Chaos, Á., Aldana, M., Benítez, M., Cortes-Poza, Y., Espinosa-Soto, C., Hartasánchez, D.A., Lotto, R.B., Malkin, D., Santos, G.J.E., et al.: Floral morphogenesis: stochastic explorations of a gene network epigenetic landscape. Plos one 3(11), e3626 (2008)
3. Banzhaf, W.: Artificial regulatory networks and genetic programming. In: Riolo, R., Worzel, B. (eds.) Genetic Programming Theory and Practice, Genetic Programming Series, vol. 6, pp. 43–61. Springer US (2003), http://dx.doi.org/10.1007/978-1-4419-8983-3_4
4. Cury, J.E., Baldissera, F.L.: Systems biology, synthetic biology and control theory: a promising golden braid. Annual Reviews in Control 37(1), 57–67 (2013)

5. Cussat-Blanc, S., Harrington, K., Pollack, J.: Gene regulatory network evolution through augmenting topologies. IEEE Transactions on Evolutionary Computation 19(6), 823–837 (2015)
6. Davidich, M.I., Bornholdt, S.: Boolean network model predicts cell cycle sequence of fission yeast. PloS one 3(2), e1672 (2008)
7. Drossel, B.: Random boolean networks. In: Schuster, H.G. (ed.) Reviews of nonlinear dynamics and complexity, pp. 69–110 (2008), http://dx.doi.org/10.1002/9783527626359.ch3
8. Fornasini, E., Valcher, M.E.: Recent developments in boolean networks control. Journal of Control and Decision 3(1), 1–18 (2016)
9. François, P., Hakim, V.: Design of genetic networks with specified functions by evolution in silico. Proceedings of the National Academy of Sciences of the United States of America 101(2), 580–585 (2004)
10. Garcia-Bernardo, J., Eppstein, M.J.: Evolving modular genetic regulatory networks with a recursive, top-down approach. Systems and synthetic biology 9(4), 179–189 (2015)
11. Goudarzi, A., Teuscher, C., Gulbahce, N., Rohlf, T.: Emergent criticality through adaptive information processing in boolean networks. Physical review letters 108(12), 128702 (2012)
12. Huang, S., Eichler, G., Bar-Yam, Y., Ingber, D.E.: Cell fates as high-dimensional attractor states of a complex gene regulatory network. Physical review letters 94(12), 128701 (2005)
13. Huang, S., Ernberg, I., Kauffman, S.: Cancer attractors: a systems view of tumors from a gene network dynamics and developmental perspective. In: Seminars in cell & developmental biology. vol. 20, pp. 869–876. Elsevier (2009)
14. Kauffman, S.A.: The origins of order: Self organization and selection in evolution. Oxford university press (1993)
15. Klamt, S., Saez-Rodriguez, J., Lindquist, J.A., Simeoni, L., Gilles, E.D.: A methodology for the structural and functional analysis of signaling and regulatory networks. BMC bioinformatics 7(1), 56 (2006)
16. Li, F., Long, T., Lu, Y., Ouyang, Q., Tang, C.: The yeast cell-cycle network is robustly designed. Proceedings of the National Academy of Sciences of the United States of America 101(14), 4781–4786 (2004)
17. Lienert, F., Lohmueller, J.J., Garg, A., Silver, P.A.: Synthetic biology in mammalian cells: next generation research tools and therapeutics. Nature Reviews Molecular Cell Biology 15(2), 95–107 (2014)
18. Lones, M.A.: Computing with artificial gene regulatory networks. In: Iba, H., Noman, N. (eds.) Evolutionary Algorithms in Gene Regulatory Network Research, Wiley, pp. 398–424 (2016), http://dx.doi.org/10.1002/9781119079453.ch15
19. Lones, M.A., Turner, A.P., Fuente, L.A., Stepney, S., Caves, L.S.D., Tyrrell, A.M.: Biochemical connectionism. Natural Computing 12(4), 453–472 (2013), http://dx.doi.org/10.1007/s11047-013-9400-y
20. Mendoza, L., Thieffry, D., Alvarez-Buylla, E.R.: Genetic control of flower morphogenesis in arabidopsis thaliana: a logical analysis. Bioinformatics (Oxford, England) 15(7), 593–606 (1999)
21. Mendoza, L., Xenarios, I.: A method for the generation of standardized qualitative dynamical systems of regulatory networks. Theoretical Biology and Medical Modelling 3(1), 13 (2006)
22. Motter, A.E.: Networkcontrology. Chaos: An Interdisciplinary Journal of Nonlinear Science 25(9), 097621 (2015)

23. Noman, N., Monjo, T., Moscato, P., Iba, H.: Evolving robust gene regulatory networks. PloS one 10(1), e0116258 (2015)
24. Reil, T.: Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In: Proceedings of the 5th European Conference on Artificial Life (ECAL'99), number 1674 in Lecture Notes in Artificial Intelligence. pp. 457–466 (1999)
25. Roli, A., Manfroni, M., Pinciroli, C., Birattari, M.: On the design of boolean network robots. In: Applications of Evolutionary Computation, Proceedings of the 2011 International Conference on Applications of Evolutionary Computation - Volume Part I, EvoApplications'11, pp. 43–52. Springer, Berlin Heidelberg (2011)
26. Saadatpour, A., Albert, R.: Boolean modeling of biological regulatory networks: a methodology tutorial. Methods 62(1), 3–12 (2013)
27. Sanchez, S., Cussat-Blanc, S.: Gene regulated car driving: using a gene regulatory network to drive a virtual car. Genetic Programming and Evolvable Machines 15(4), 477–511 (2014), http://dx.doi.org/10.1007/s10710-014-9228-y
28. Singh, V.: Recent advances and opportunities in synthetic logic gates engineering in living cells. Systems and synthetic biology 8(4), 271–282 (2014)
29. Taou, N.S., Corne, D.W., Lones, M.A.: Evolving boolean networks for biological control: State space targeting in scale free boolean networks. In: Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2016 IEEE Conference on. pp. 1–6 (2016), http://dx.doi.org/10.1109/CIBCB.2016.7758125
30. Taou, N.S., Corne, D.W., Lones, M.A.: Towards intelligent biological control: Controlling boolean networks with boolean networks. In: European Conference on the Applications of Evolutionary Computation. pp. 351–362 (2016), https://doi.org/10.1007/978-3-319-31204-0_23
31. Timmis, J., Alden, K., Andrews, P., Clark, E., Nellis, A., Naylor, B., Coles, M., Kaye, P.: Building confidence in quantitative systems pharmacology models: An engineer's guide to exploring the rationale in model design and development. CPT: pharmacometrics & systems pharmacology 6(3), 156–167 (2017)
32. Trefzer, M.A., Kuyucu, T., Miller, J.F., Tyrrell, A.M.: Evolution and analysis of a robot controller based on a gene regulatory network. In: Tempesti, G., Tyrrell, A.M., Miller, J.F. (eds.) Evolvable Systems: From Biology to Hardware, Lecture Notes in Computer Science, vol. 6274, pp. 61–72. Springer Berlin Heidelberg (2010), http://dx.doi.org/10.1007/978-3-642-15323-5_6
33. Trefzer, M.A., Kuyucu, T., Miller, J.F., Tyrrell, A.M.: Image compression of natural images using artificial gene regulatory networks. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation. pp. 595–602. GECCO '10, ACM, New York, NY, USA (2010), http://dx.doi.org/10.1145/1830483.1830593
34. Turner, A.J., Miller, J.F.: Recurrent cartesian genetic programming of artificial neural networks. Genetic Programming and Evolvable Machines 18(2), 185–212 (2017)
35. Veliz-Cuba, A., Arthur, J., Hochstetler, L., Klomps, V., Korpi, E.: On the relationship of steady states of continuous and discrete models arising from biology. Bulletin of mathematical biology 74(12), 2779–2792 (2012)