Modularity Within Artificial Gene Regulatory Networks

1st George Lacey
Computer Science
University of Hull
Hull, UK
G.A.Lacey@2014.hull.ac.uk

2nd Annika Schoene
Computer Science
University of Hull
Hull, UK
A.M.Schoene@2017.hull.ac.uk

3rd Nina Dethlefs
Computer Science
University of Hull
Hull, UK
n.dethlefs@hull.ac.uk

4th Alexander Turner
Computer Science
University of Hull
Hull, UK
alexander.turner@hull.ac.uk

***Abstract*—Modularity is a feature of found in biological systems where it is common for functionally related processes to evolve to be individually discrete units. Such traits are prevelant in prokaryotic genomes. This work aims to understand to what extent artificial gene regulatory networks AGRNs, which take inspiration from gene regulation in nature will self-divide into modular task specific sub-networks consisting of multiple interacting nodes when solving multiple complex tasks. To investigate this, we evolve AGRNs to solve three different tasks with ranging dynamics simultaneously and evaluate the network structure. From this we aim to build an understanding of whether modularity in AGRNs is fundamental to solving multiple tasks and what effect the nature of the tasks being solved has on modularity within the networks.**

***Index Terms*—gene regulation, evolutionary algorithms, modularity**

I. Introduction

Gene regulatory networks are fundamental to the homeostatic regulation of cellular functions. Biological cells exhibit properties such as robustness and adaptability which has resulted in many disciplines attempting to capture and model these behaviours and traits insilico [1], [29]. There are two principle reasons for this. The first is to experiment with this systems to better understand their behaviour and biological underpinnings [2], [3], [10]. The second is to exploit these features to build effective and robust machine learning techniques, typically for the solving of complex tasks and the analysis of complex systems [9], [10], [13]–[15]. This work focuses on the latter.

Gene in nature which are related in function often occur close to each other in the genome. This is because, particularly in prokaryotic gene regulation, it is often more efficient to transcribe related genes all at once. The close proximity of genes in this way is know as gene linkage or gene clustering [7]–[10]. Because of this, such structures are often preserved from an evolutionary perspective as they are both beneficial and efficient.

In this work, we consider how artificial gene regulatory networks (AGRNs) evolve to solve multiple complex problems, and how the structure of the networks changes in reference to modularity, and if this is dependent of the task being solved. In terms of modularity, we are specifically looking at to what degree nodes within the AGRN segregate themselves depending on the task for which they are applied, and wether nodes are shared between tasks. We will use three tasks for which the AGRNs will be applied. The pendulum cart task, the double pendulum task and the mountain car task. All of these taks are part of the openAI gym package [11], designed to test artificial intelligence techniques.

II. Biological gene regulation

Gene regulation in nature is the process of either inducing or repressing the expression of a gene. A gene is a region of DNA for which a functional genetic unit can be transcribed. Transcription is the process of copying a segment of DNA into an RNA sequence, and is typically the first process of gene regulation. A gene can be thought of as a unit of heredity. An example of gene regulation can be found within the functionality of operons, which are sections of DNA that allow for the transcription of a set of genes that are usually related in their functionality. The operon can be itself considered a functional unit of modularity. RNA polymerase binds to the promoter region of an operon in order to transcribe the gene(s); however, transcription may be inhibited by the operator of the operon if bound to by a transcription factor known as a repressor. Transcription factors are proteins that control the transcription of genes, such as activators that bind to a site next to the promoter in order to increase the rate of transcription [24]–[26]. For homeostasis to occur within an organism, the macro molecules associated with gene expression must be finely regulated.

There are many different processes which typically regulate gene expression [34]–[37]. One of the most prominent is tanscriptional modifications, such as the prevention of binding of transcription factors. Other such modifications are post-transcriptional modifications and translation modification. Although these processes are complex, gene expression is fundamentally robust to perturbations, maintaining homeostasis under a wide range of varying conditions [27]. One example of this is the E. coli bacterium, where the lactose operon allows the bacterium to use lactose as a fuel source when it is present and glucose is not [27], [28].

A genetic network can be considered a weighted graph, where each nodes is a gene, and the weight is a product of the influence one gene has upon another. Gene regulation possess

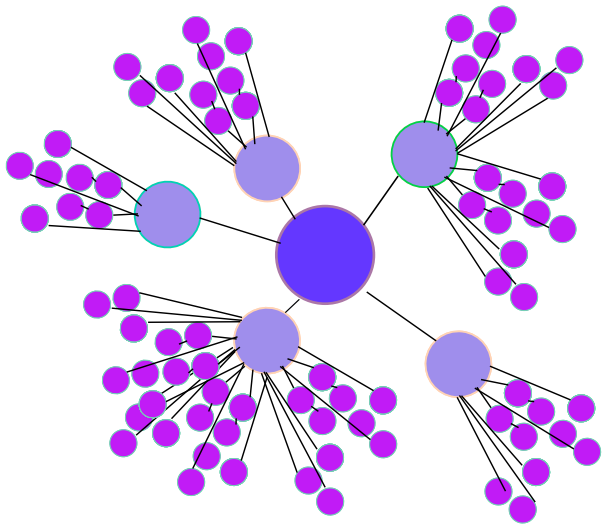

Fig. 1. An abstraction form [12] which shows modularity in the yeast protein network, for which many of the genes located close together are functionally related.

some of the properties which would be advantageous to abstract over to insilico devices. Their adaptability, robustness and stability over time make this process an idea candidate to study from a computational perspective, specifically for the control of complex systems. In addition, because their structure can be abstracted down to a weighted graph, this can be simply incorporated into a computational system.

A. Evolution

Evolution is the process of change over time, and from a biological perspective is the heritable adaptation of an organism over successive generations [30], [31]. Biological evolution is focused on the DNA molecule which holds the information in which to code for proteins and other macro molecules used by a cell. Evolution to an extent exploits the robustness of gene regulation to allow for modification to produce meaningful, but non lethal change [1], [16]–[18]. That is, in general and organism can have its DNA modified without fatal consequences. Modularity is one of the features which makes this possible [23]. In addition, redundancy is a feature where genes or operons are encoded by more than one DNA segment, therefore if one becomes damaged, the other can maintain functionality. Another feature is decoupling, which specifies that the phenotype of an organism is not directly encoded but its a specified by an intermediary which is the type frequency of macro molecules in the cell provided by the transcription of the genome.

For the AGRNs to evolve well, they must abstract in part some of the appropriate properties of gene regulation in biology to ensure robustness throughout the optimisation processes [32], [33].

III. Artificial Gene Regulatory Networks

Artificial gene regulatory networks (AGRNs) are computational models that take inspiration from biological gene regulation and are designed for problem solving. They have been shown to be particularly adept at solving complex time-series tasks [19]–[21]. They consist of a set of indexed nodes, each with a parameterised regulatory function which is typically a sigmoid function (Figure 2). The parameters can be seen in Table III. The networks state is a function of the nodes underlying expression levels, which are real number values in the range of 0-1.

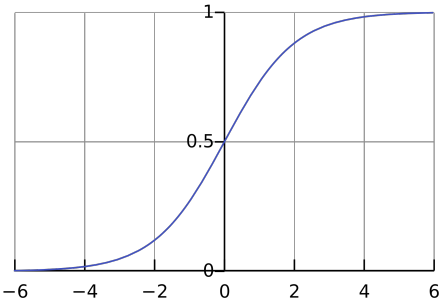

Fig. 2. The sigmoid function which is at the core of each gene. The slope can be modified by the slope and offset parameters in the genes to change the characteristics and regulatory function of the genes

In this work the connectivity of the nodes within the AGRN are represented using 2-dimensional virtual space, with each node having a position defined by its position and length variables shown in Table III. The 'Position' parameters determine the position of the centre point of the node, and the 'Length' parameters determine the extent to which each node's space extends from the centre point into the appropriate dimensions. If the areas of two genes overlap, then they are connected. An example of node connectivity is shown in Figure. 3.

Algorithm 1 displays how AGRNs execute. During the first iteration the nodes are initialised by setting their parameters to random values within the corresponding ranges (Table III). At each iteration the task observations are mapped onto the input nodes (replacing the random values in the first iteration), where the task observations are transformed to valid expression levels if required. The expression levels of the nodes are then calculated based on the expression levels from the previous iteration. The expression levels of the output nodes from the network are used to control the tasks, by mapping and transforming them to a value within the ranges of that specified task. The AGRN's behaviour is an emergent property of the functionality of its underlying genes.

Formally, this AGRN architecture can be defined by the tuple $\langle G, L, \mathsf{In}, \mathsf{Out} \rangle$, where:

G is a set of genes $\{n_0 \dots n_{|N|} : n_i = \langle a_i, I_i, W_i \rangle\}$ where:

$a_i : \mathbb{R}$ is the activation level of the gene.

$I_i \subseteq G$ is the set of inputs used by the gene.

W_i is a set of weights, where $0 \leq w_i \leq 1$, $|W_i| = |I_i|$.

L is a set of initial activation levels, where $|L_N| = |N|$.

$\mathsf{In} \subset G$ is the set of genes used as external inputs.

$\mathsf{Out} \subset G$ is the set of genes used as external outputs.

$$f(x) = (1 + e^{-sx-b})^{-1} \quad (1)$$

$$x = \sum_{j=1}^{n} i_j w_j \quad (2)$$

Name	Type	Range
Expression	Real	[0,1]
Weight	Real	[0,1]
Slope	Real	[0,20]
Sigmoid offset	Real	[0,1]
Position X	Real	[0,1]
Position Y	Real	[0,1]
Length X	Real	[0,0.5]
Length Y	Real	[0,0.5]

TABLE I
THE VARIABLES HELD WITHIN EACH GENE WITHIN THE AGRN. THE WEIGHT, SLOPE AND OFFSET ARE USED IN THE FUNCTION (1) (FIGURE 2) IN WHICH TO DEDUCE A GIVEN GENES' EXPRESSION. THE POSITION AND LENGTH VARIABLES DEDUCE WHICH GENES ARE CONNECTED TO WHICH BASED UPON THE PROXIMITY OF THESE VALUES.

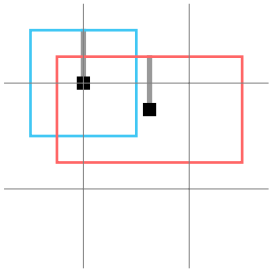

Fig. 3. Illustration of two connected nodes. The outer blue and red rectangles represent the whole nodes, the inner black rectangles represent the centre point of each node. The centre point of the blue node exists within the space of the red node, meaning that the blue node acts as an input to the red node.

Algorithm 1 Execute AGRN on a task

```
for x = 1 → Number of iterations do
    if x is 1 then
        Initialise nodes randomly
    end if

    Scale task observations to 0,1
    Map task observations to input nodes

    for x to numberofnodes do
        Initialise nodes randomly
    end for

    Scale outputs to task range
    Map output nodes to task actions
end for
```

IV. TASKS

The AGRNs will be applied to solve three independent tasks within the same network. The focus of this work is to look at the modularoty and the connectivity of AGRNs which can solve multiple tasks. To achieve this, we use three different tasks which require a range of dynamics to solve completely and will all be independent. All of these tasks are from the openAi gym [11].

A. Pendulum cart task

The pendulum task shown in Figure 4 is a real time control task. The objective of the task is to move the cart in a 1-dimensional space to ensure that the pendulum remains upright. The movement of the pendulum must be monitored in order to enact counter-movements to stabilise it. The movement of the cart is controlled using two discrete actions shown in Table II, one which moves it left, and one which moves it right at a constant speed. The four observations of the task are taken from the position and velocity of the cart, as well as the angle and velocity of the outward end of the pendulum as shown in Table III. If the pendulum pivots too far in either direction, the cart moves too far in either direction, or the task runs for 500 time steps it will cease execution. The reward is determined by the duration the task remains active.

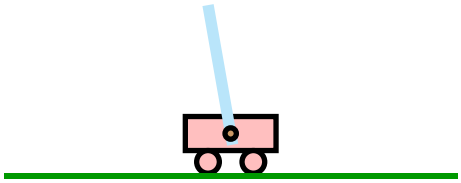

Fig. 4. Illustration of pendulum cart task. The pendulum has pivoted counterclockwise from the upright position. The cart could be moved to the left to correct this. The overall objective of this tasks is to maintain the pendulum in the upright position for as long as possible.

TABLE II
ACTIONS OF THE PENDULUM CART TASK

Name	Type	Values
Cart control	Discrete	0 (Left), 1 (Right)

TABLE III
OBSERVATIONS OF THE PENDULUM CART TASK

Name	Type	Min	Max
Cart position	Real	0	1
Cart velocity	Real	-10	10
Pole angle	Real	0	360
Pole velocity	Real	-20	-20

B. Double pendulum task

The double pendulum task, shown in Figure 5 is another dynamic control task. A double pendulum pivots around a fixed point, and by controlling only the second joint the double

pendulum must be moved so that the outer end of the outer pendulum reaches a fixed height. The height is represented by the dark blue horizontal line in the illustration.

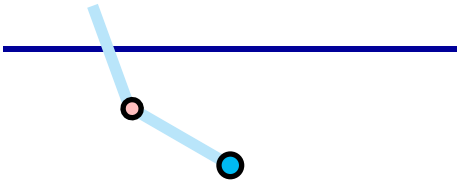

Fig. 5. Illustration of the double pendulum task. In this instance, the outer pendulum has reached the height indicated by the horizontal line so the task has been completed. The reward of this task is related to the amount of time that is taken to achieve this

TABLE IV
ACTIONS OF THE DOUBLE PENDULUM TASK

Name	Type	Values
Torque	Discrete	-1 (Negative torque), 0 (No torque), 1 (Positive torque)

TABLE V
OBSERVATIONS OF DOUBLE PENDULUM TASK

Name	Type	Min	Max
Cosine of first joint angle	Real	-1	1
Sine of first joint angle	Real	-1	1
Cosine of second joint angle	Real	-1	1
Sine of second joint angle	Real	-1	1
First joint velocity	Real	-4π	4π
Second joint velocity	Real	-9π	9π

The movement of the double pendulum is controlled a single discrete action, shown in Table IV. Torque may be applied to the second joint via this action causing the outer pendulum to swing clockwise or counter-clockwise.

The six observations of the task, shown in Table V are taken from the sine and cosine of the angles of the two joints as well as the rotational velocity of the joints. The reward is determined by the time it takes for the pendulum to reach the fixed height, and the task will stop after 500 time steps.

C. Mountain car task

The mountain car task, shown in Figure 6 consists of a car that must gain enough momentum to reach the goal, at the top of a mountain. The car is not powerful enough to accelerate straight up to the goal, so must travel backwards and forwards in order to gain enough momentum. The movement of the car is controlled a single discrete action, shown in Table VI. The car can accelerate left, right, or not at all via the action.

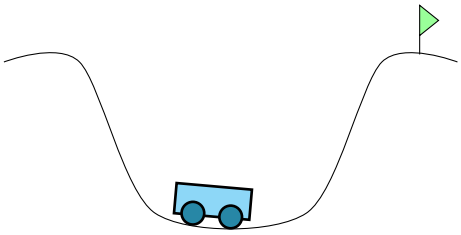

Fig. 6. Illustration of the mountain car task. The car is at its starting position between the two mountains. The car must gain momentum to reach the goal indicated by the green flag at the summit of the mountain on the right.

TABLE VI
ACTIONS OF MOUNTAIN CAR TASK

Name	Type	Values
Torque	Discrete	0 (Move left), 1 (Don't move), 2 (Move right)

The two observations of the task, as shown in Table VII are taken from the position and velocity of the car. The reward is based on the progress of the car, the closer the car gets to the goal, the higher the reward will be. The reward is subject to penalisation based on the duration the task runs for, and the reward is proportional to the amount of time taken to complete the task.

D. Mapping tasks to AGRN

The AGRNs in this paper will be trained to solve three completely independent tasks simultaneously. Each task's environment provides observations and a means for the AGRN to evoke an action. Each observation is mapped onto a separate input node chosen randomly before the network starts evolving. The expression of the input nodes is set to the observation of the respective task in which it is an input for. Observations that are not already represented by real numbers ranging between 0 and 1 are normally transformed to fit these criteria. Similarly, a randomly selected node from the network will act as the network output, for which the expression of that gene will be mapped back to the environment. The observation and output mapping is done randomly once at the start of the experimentation, and once assigned, they do not change throughout the optimisation process.

V. EXPERIMENTATION

A mutation only genetic algorithm (Algorithm 2) will be used to evolve the AGRNs with a population size of 1000. This is to provide a level of transparency over the optimisational process which allows the inspection and analysis of a

TABLE VII
OBSERVATIONS OF MOUNTAIN CAR TASK

Name	Type	Min	Max
Position of car	Real	-1.2	0.6
Velocity of car	Real	-0.07	0.07

specific AGRN during optimsation. If crossover was permitted, significant parts of the network would be recombined, and the genotyoe to phenotype mapping would be difficult to analyze over just a few generations. Moreover, in this work we are not motivated by objective performance of the networks or the optimsation process, as long as the networks are able to solve the three tasks. Hence, the trade off between potential loss of speed and performance is beneficial for the tractability of the evolutionary process. During the mutation operation, the parameters of each node within the network have a chance of being set to a random values according to a normal distribution centered around the current value. (or 'mutated'). At each iteration of the algorithm, the best 10 will be immediately copied to the child population. This concept is known as 'elitism' and guarantees that the best networks will not be lost due to random chance. The fitness of a give AGRN will be the sum of its performance on all three tasks.

Algorithm 2 Execute mutation only genetic algorithm

$P \leftarrow \{\}$ {Initialise empty initial population}

for $x = 1 \rightarrow$ Population size **do**
  $P \leftarrow P \cup$ Randomly initialised AGRN
end for

for $y = 1 \rightarrow$ Number of generations **do**
  for all $p \in P$ **do**
    EVALUATE(p)
  end for

  $Q \leftarrow \{\}$ {Initialise empty child population}
  $Q \leftarrow Q \cup$ Elite members
  repeat
    $Q \leftarrow Q \cup$ TOURNAMENT_SELECT(P)
  until $|Q|$ is Population size
  $P \leftarrow Q$

  Mutate population
end for

Only AGRNs that solve all 3 tasks are useful for testing, so networks that do not fit this criteria will not be analysed in this work. In total, each network has 18 nodes, consisting of 12 input nodes, 4 for the pendulum cart task, 6 for the double pendulum task and 2 for the mountain car task. There will be three output nodes, one for each tasks, as well as 3 general processing nodes for the AGRNs to use to process the information.

Networks often evolve to function without utilising all of their nodes,in these cases it is beneficial to remove redundant nodes in order to simplify analysis. This process is shown in Algorithm 3, where each node is disabled in turn, and if the performance does not decrease the node is effectively removed from the network by preventing its expression level from affecting other nodes.

Algorithm 3 Purge nodes from network

$t \leftarrow$ Threshold
$b \leftarrow$ Simulate network {Base performance}
$N \leftarrow$ Network nodes

for all $n \in N$ **do**
  $N \leftarrow N \setminus \{n\}$ {Remove node from network}
  $p \leftarrow$ Simulate network {Performance without node}
  if $p < b - t$ **then** {Performance decreases}
    $N \leftarrow N \cup \{n\}$ {Replace node}
  end if
end for

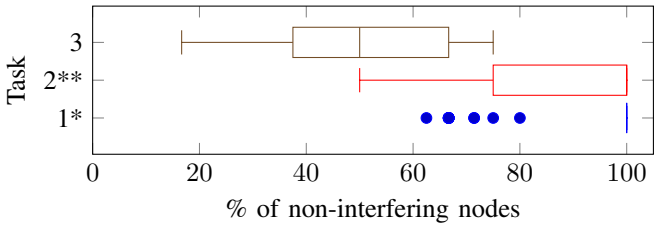

Fig. 7. Box plot of the percentage of non-interfering nodes of each task within the networks. *All values are equal to 1 except outliers. **Median and upper quartile are equal to 1. It can be seen that there is a clear difference between the three tasks in reference to the amount of interfering nodes within the AGRNs that solved them. This indicates that modularity within the tasks may be a result of they dynamics and complexity of the task its solving.

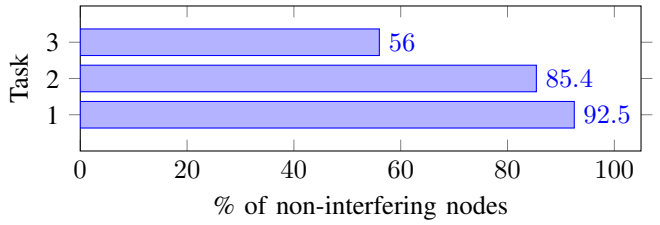

Fig. 8. Bar plot of the average percentage of interfering nodes based on task. This data is a transformation of the data found in Figure 7, and clearly highlights the difference in non-interfering nodes between the three tasks.

VI. Results

The networks were trained to solve three independent tasks:

- Task 1 - The pendulum cart
- Task 2 - The double pendulum
- Task 3 - The mountain car

The networks will be analysed based on how they interact with each task. Nodes from the network that were mapped onto the task observations or task actions are said to belong to that task. Nodes that do not belong to any task are referred to as 'processing nodes', as no external value is mapped onto or set from them. The connections between nodes were analysed in order quantify the clustering of the networks. Each task has a separate action, mapped from an action node within a network. Therefore, if a node connects directly to the corresponding action node, it must influence the task in some

way. Additionally, nodes that indirectly connect to the action node via a connecting node may influence the task.

In order to show how networks internally organise themselves, the nodes connecting to each action node were split into three groups: the observation nodes belonging to that task, processing nodes, and nodes belonging to other tasks. Nodes belonging to other tasks will be referred to as 'interfering nodes', as it is unlikely that their influence is beneficial as their expression level is not based on the task in question. The exception to this is an action node that could be influenced by an observation node belonging to the correct task; however, because all of the tasks must be solved to some degree it is likely that action nodes in this position will also be influenced by an observation node belonging to it's own task, and will therefore cause interference anyway.

For each network, the percentage of non-interfering nodes connected to the action node of each task was calculated. This is simply the percentage of connecting nodes that are observation nodes (belonging to the same task) or processing nodes. The results are plotted in Figure 7 and the averages in Figure 8. On the surface, the results show that networks typically do organise themselves based on the three tasks as expected, where tasks one and two in particular have a low number of interfering nodes on average, this is also true for task three but to a lesser degree.

There are multiple possible reasons for the discrepancy amongst tasks, it could be the case that interference affects simpler tasks less, as they have simpler dynamics so do not require as much control. This would make sense in this case as the mountain car task is the simplest task as it only requires basic movement, and in theory can be solved solely based on the position of the car. The pendulum cart task had a much lower amount of interference on average and is a more complicated task, as the angle and velocity of the pendulum must be monitored so that it can be stabilised, the position of the cart must also be monitored so that it does not go out of bounds. Similarly, the double pendulum task requires that the angles and velocities of the two pendulum are monitored so that the correct swinging motion can be enacted in order to move it to the correct position to solve the task. Each task provides a reward that represents how successful the networks are at completing it. The means of calculating this differs amongst the tasks, and the total pendulum cart reward is calculated as the duration that the pendulum remains balanced, whereas the other tasks are penalised based on how long it takes for the task to be completed. It is possible to receive the maximum reward for the pendulum cart task, but infeasible for the others. This could result in the networks favoring the tasks with a lower realistic upper reward bound.

A network with no interference is shown in Figure 9, and the connections in Table VIII. The nodes acting as input to the task action nodes are all observation nodes belonging to the same task, and the processing nodes connecting to task nodes do not take any input from other tasks. Whilst this network would be much easier to analyse, complex networks are highly unlikely to develop in this way, as its subtasks will not be completely independent. A network with high levels of interference is shown in Figure 10, and the connections in Table IX, despite the interference, the network performs as well as the network with no interference due its robustness.

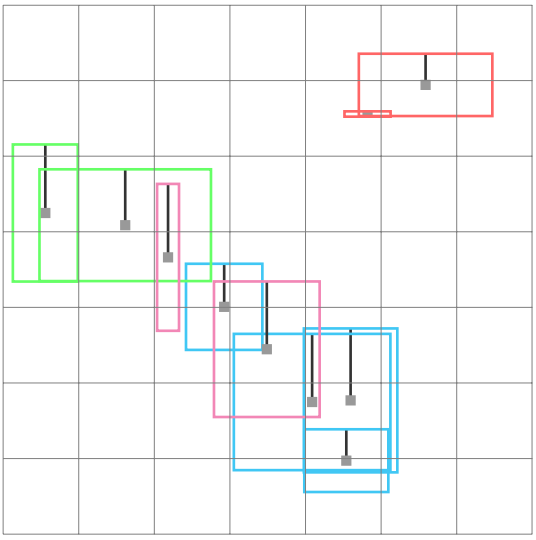

Fig. 9. Layout of network with no interference, each coloured rectangle represents a node belonging to a specific task. Pink nodes belong to no task.

TABLE VIII
DISPLAY OF NODE CONNECTIONS FOR NETWORK WITH NO INTERFERENCE.

Task	Task nodes	Processing nodes	Other task nodes
1	0, 2, 3	16	none
2	8, 9	none	none
3	11	15, 17	none

The performance of networks has been plotted against the average percentage of non-interfering nodes in Figure 11. It might be assumed that networks with less interference perform better; however, this is not the case. There are networks that perform well and poorly with a range of interference.

TABLE IX
DISPLAY OF NODE CONNECTIONS FOR NETWORK WITH HIGH INTERFERENCE.

Task	Task nodes	Processing nodes	Other task nodes
1	2, 3	15, 16, 17	9, 10, 13
2	9	none	10
3	10, 11	none	0, 9, 13

VII. CONCLUSION

In this paper, we optimised artificial gene regulatory networks (AGRNs) to solve multiple independent tasks to understand if the AGRNs adopt modulatory and task independence within the networks.

We have shown how AGRNs internally organise their nodes in this situation, and found that networks evolve with a variety of structures for which there is no specific rule. However,

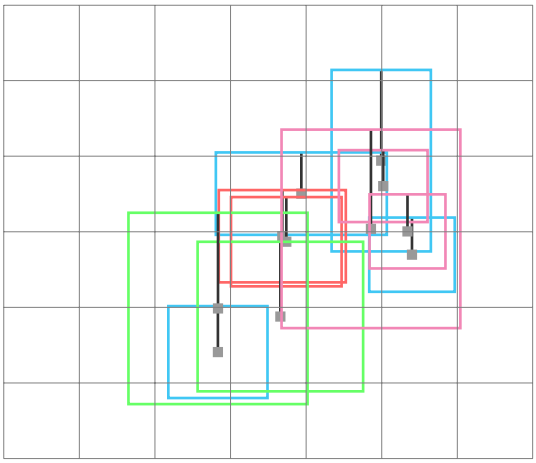

Fig. 10. Layout of network with one of the highest rates of interference, each coloured rectangle represents a node belonging to a specific task. Pink nodes belong to no task.

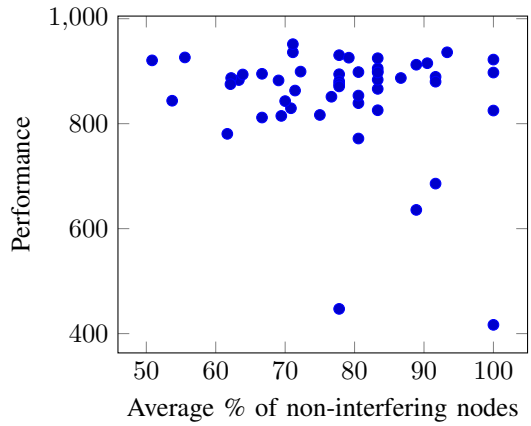

Fig. 11. Scatter plot of the affects of interference on performance.

one key result from this work is that it is clear that AGRNs organise their structure during optimisation according in part to the nature of the tasks they are solving, with the mountain car task generally having a significantly lower percentage of interacting nodes than other tasks.

In general, some networks evolved with clear separation between nodes working towards different tasks; however, in most cases there is at least some interference, where nodes working towards completely independent tasks interact with each other.It is likely that crossover would be more common when solving complex tasks, as the sub-tasks may share inputs (observation nodes) and even functionality. We also found that interference is not indicative of performance, most likely due to the robustness of networks, meaning that there is not necessarily an incentive for networks to organise themselves into clear sub-networks.

It is clear from this work that due to the nature of AGRNs, it is unlikely for networks to organise themselves into completely independent modules. Further research will have to be conducted in order to analyse the networks in this circumstance, and determine whether it is feasible to recognise sub-networks despite interference from potentially unrelated nodes.

References

[1] H. Kitano, "Biological robustness", Nature Reviews Genetics, vol. 5, pp. 826–837, November 2011.

[2] Herman F Fumia and Marcelo L Martins. Boolean network model for cancer pathways: predicting carcinogenesis and targeted therapy outcomes. *PloS one*, 8(7):e69008, 2013.

[3] Stuart Kauffman, Carsten Peterson, Björn Samuelsson, and Carl Troein. Random boolean network models and the yeast transcriptional network. *Proceedings of the National Academy of Sciences*, 100(25):14796–14799, 2003.

[4] Rui-Sheng Wang, Assieh Saadatpour, and Reka Albert. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology*, 9(5):055001, 2012.

[5] David Snyder, Alireza Goudarzi, and Christof Teuscher. Finding optimal random boolean networks for reservoir computing. *Artificial Life*, 13:259–266, 2012.

[6] Rui-Sheng Wang, Assieh Saadatpour, and Reka Albert. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology*, 9(5):055001, 2012.

[7] Jeffrey G Lawrence. Shared strategies in gene organization among prokaryotes and eukaryotes. *Cell*, 110(4):407–413, 2002.

[8] Anne E Osbourn and Ben Field. Operons. *Cellular and Molecular Life Sciences*, 66(23):3755–3775, 2009.

[9] David Snyder, Alireza Goudarzi, and Christof Teuscher. Finding optimal random boolean networks for reservoir computing. *Artificial Life*, 13:259–266, 2012.

[10] Rui-Sheng Wang, Assieh Saadatpour, and Reka Albert. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology*, 9(5):055001, 2012.

[11] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. and Zaremba, W., OpenAI gym. arXiv preprint arXiv:1606.01540, 2016.

[12] E Zotenko, J Mestre, DP O'Leary, and TM Przytycka. 'Why do hubs in the yeast protein interaction network tend to be essential: Reexamining the connection between the network topology and essentiality'. PLoS Comput Biol, 4(8):e1000140 2008.

[13] M. A. Lones, "Controlling complex dynamics with artificial biochemical networks', in Genetic Programming, Istanbul, Turkey, pp. 159–170, 2010.

[14] A. P. Turner et al., "Using artificial epigenetic regulatory networks to control complex tasks within chaotic systems", in Information Processing in Cells and Tissues, Cambridge, UK, pp. 1–11, 2012.

[15] T. Quick, C. L. Nehaniv, K. Dautenhahn, and G. Roberts, "Evolving embodied genetic regulatory network-driven control systems", in European Conference on Artificial Life, Dortmund, Germany, pp. 266–277, 2003.

[16] Aderem, A., 'Systems biology: its practice and challenge's. Cell, 121(4), pp.511-513 2005

[17] Csete, M.E. and Doyle, J.C., Reverse engineering of biological complexity', science, 295(5560), pp.1664-1669, 2002

[18] Pigliucci, M., 'Is evolvability evolvable?'. Nature Reviews Genetics, 9(1), p.75, 2008

[19] T. Taylor, "A genetic regulatory network-inspired real-time controller for a group of underwater robots", in Conference on Intelligent Autonomous Systems, Amsterdam, Netherlands, pp. 403–412, 2004.

[20] A. P. Turner et al., "Using epigenetic networks for the analysis of movement associated with levodopa therapy for Parkinson's disease", Biosystems, vol. 146, pp. 35–42, 2016.

[21] S. Das, P. Koduru, X. Cai, S. Welch, and V. Sarangan, "The gene regulatory network: an application to optimal coverage in sensor networks", in GECCO, Atlanta, GA, USA, pp. 1461–1468, 2008.

[22] H. A. Simon, "The architecture of complexity", Proceedings of the American Philosophical Society, vol. 106, pp. 467–482, 1962.

[23] H. Lipson, "Principles of modularity, regularity, and hierarchy for scalable systems", Journal of Biological Physics and Chemistry, vol. 7, pp. 125–128, 2007.

[24] B. L. Wanner, "Gene regulation by phosphate in enteric bacteria", Journal of Cellular Biochemistry, vol. 51, pp. 47–54, 1993.

[25] J. François, D. Perrin, C. Sánchez, and J. Monod, “The operon: a group of genes whose expression is co-ordinated by an operator”, Compte Rendu de l’Academie des Sciences, vol. 250, pp. 1727–1729, 1960.
[26] M. Ptashne and A. Gann, “Genes and Signals”. Cold Spring Harbor, New York: Cold Spring Harbor Laboratory Press, 200, p. 2.
[27] F. Jacob and J. Monod, “Genetic regulatory mechanisms in the synthesis of proteins”, Journal of Molecular Biology, vol. 3, pp. 318–356, 1961.
[28] L.A. Moran, H.R. Horton, K.G. Scrimgeour, and M.D. Perry, Principles of Biochemistry, 5th ed. USA: Pearson, 2011, pp. 650–653.
[29] S. A. Kauffman, “Metabolic stability and epigenesis in randomly constructed genetic nets”, Journal of Theoretical Biology, vol. 22, pp. 437–467, 1969.
[30] Pigliucci, M., Murren, C.J. and Schlichting, C.D., Phenotypic plasticity and evolution by genetic assimilation. Journal of Experimental Biology, 209(12), pp.2362-2367. 2006
[31] Hansen, T.F., ÁLvarez-Castro, J.M., Carter, A.J., Hermisson, J. and Wagner, G.P., Evolution of genetic architecture under directional selection. Evolution, 60(8), pp.1523-1536. 2006
[32] Schlosser, G. and Wagner, G.P. eds., 2004. Modularity in development and evolution. University of Chicago Press.
[33] Clune, J., Mouret, J.B. and Lipson, H.,The evolutionary origins of modularity. Proc. R. Soc. B, 280(1755), p.20122863. 2013.
[34] Mou, Shaoshuai, and Domitilla Del Vecchio. ”Transcription factor loads tend to decrease the robustness of stable gene transcription networks.” bioRxiv (2016):
[35] Zhao, Boxuan Simen, Ian A. Roundtree, and Chuan He. ”Post-transcriptional gene regulation by mRNA modifications.” Nature reviews Molecular cell biology 18, no. 1 (2017)
[36] Tatusova, Tatiana, Michael DiCuccio, Azat Badretdin, Vyacheslav Chetvernin, Eric P. Nawrocki, Leonid Zaslavsky, Alexandre Lomsadze, Kim D. Pruitt, Mark Borodovsky, and James Ostell. ”NCBI prokaryotic genome annotation pipeline.” Nucleic acids research 44, no. 14 (2016)
[37] Willbanks, Amber, Meghan Leary, Molly Greenshields, Camila Tyminski, Sarah Heerboth, Karolina Lapinska, Kathryn Haskins, and Sibaji Sarkar. ”The evolution of epigenetics: from prokaryotes to humans and its biological consequences.” Genetics and epigenetics 8 (2016)