

Bio-Inspired Multi-agent Embryonic Architecture for Resilient Edge Networks

Abstract—With the pervasive introduction of IoE technologies, use-cases are frequently being found operating within harsh environmental conditions. This decrees the need for solutions which permit service delivery to operate in a highly-resilient manner. This work presents an architecture for a novel cloud platform designed for resilient service delivery. It supports networks where poor communication links or high node failure will cause services to be delivered in a non-resilient manner. This could be the result of factors such as high-node mobility, poor environmental conditions, unreliable infrastructure from environment disaster or cyber-attack. This biologically inspired architecture uses a purely distributed multi-agent approach to provide self-healing and self-organising properties, modelled on the characteristics of embryonic development and biological cell communication. To permit high-levels of a node churn, this multi-agent approach uses local-only communication. Probabilistic Cellular Automata are used to simulate this architecture and evaluate the efficacy of this approach.

Index Terms—IoE, Embryonic, Multi-agent, Resilience, Bio-Inspired, Cloud, Edge, Artificial Life, Artificial Intelligence, Cybersecurity, Cellular Automata.

I. INTRODUCTION

In common environments where networked devices are frequently found and relied upon: the home, work-place, public areas etc, persistent service delivery is taken for granted. Minor breaks in service may subjectively appear poor but this is largely due to their infrequency. Objectively, these networks deliver their service persistently due to the high-determinism of the environments they operate within.

We refer to the term resilience as a network's ability to persistently deliver it's service within the face of various internal and external changes [1]. If these commonly found networks are not robust enough to continue operating during changes in their internal or external environment then they do not operate resiliently.

Conversely there are many types of networks which must persist in their service delivery yet their underlying environmental conditions are non-deterministic, resulting in Intermittently Connected Networks. Examples of these include: Mobile Ad-hoc networks such as with mobile devices or vehicle communications, wireless sensor networks used for environmental monitoring often in extreme conditions and Exotic Media Networks in highly disruptive locations such as space. Most solutions rely on employing delay or disruption tolerant networking approaches such as store and forward, parallel routing, error correction etc. [2].

A number of use-cases are highlighted within the application of Industrial IoE where a large number of nodes collect data on the edge [3]. Mining [4], transportation, crime

[5] and agriculture [6] are examples where large scale IoE systems may operate in environments with changing and non-deterministic environmental conditions in addition to device mobility.

Networking solutions are not compatible with all types of harsh environments. Some use-cases involve the destruction of nodes such as adversarial attacks in warfare [7] or node subversion during cyber-attack [8]. For these situations, novel architectures are often sought to enhance the network's environment. These take the form of distributed over centralised architectures, mass redundancy of resources, or diversity across hardware and software and location [9]. When seeking alternative architectures, a common source of inspiration lies within biological systems due to their inherently scalable and resilient nature. Hence this work contributes to the area of biologically inspired resilient computing architectures through the application of embyronics to the problem of IoE/ Fog operating within harsh environments.

The rest of this paper is as follows: Section 2 presents related work, section 3 describes the proposed embryonic platform as applied to fog networks, section 4 provides a description for the proposed Cellular Automata Model. Section 5 presents the simulation model with results. Section 6 presents an analysis of the results whilst section 7 concludes the work.

II. RELATED WORK / MOTIVATION

This section presents the related work for this research in the area of biologically inspired techniques for IoE/Fog resilience. Levering Fog computing nodes as a data processing platform for IoE is discussed in a number of places [10] with use-cases such as energy management [11], smart-cities [12] benefiting from simple data pre-processing on the edge nodes. However, the platform proposed in this paper aims to provide a more feature rich platform for data processing through the use of a micro-services architecture.

To the best of the authors' knowledge, the problems of resiliency within these platforms as a consequence of node mobility, hostile environments or threats to the architecture are still open. A solution to this can be found within biological-inspired techniques, which have been applied to a whole host of problems within computer science, the Turing machine which was based upon a human clerk is arguably the first. In terms of resilient networks/systems, few works exist in the area of FoE due to it's novelty, however some can be seen within similar domains. For example cellular signalling models (which operate in a message and not node oriented fashion) have also been applied within wireless ad-hoc networks for resilient and efficient communication [13]. Whilst numerous bio-inspired techniques for resilient cloud computing and other

distributed systems have been investigated [14]. These can involve architectural models similar to this work using cell-based models [15], networking approaches based upon the distributed processing capabilities of ant colonies [16], intrusion detection methods [17] or service orchestration methods [18] based upon artificial immune systems, alternatives to public key infrastructure based upon family genetics [19] and many more.

This work applies embryonics, the field of developing systems which are based upon the characteristics of embryogenesis [20]. Embryogenesis refers to the development of a biological being from a mother cell, the zygote. This biological process provides a high degree of resilience through redundancy and self-healing. Embryonic inspired electronics have shown that resilience may be achieved by modelling a system on these self-healing capabilities [21], whilst embryonic software has shown similar self-healing properties for distributed systems [22].

This work proposes an architecture which is deemed timely as a platform for combined Fog and IoE networks, also referred to as Fog-of-Everything (FoE) [10] [23]. Initially, cloud computing was deemed the supporting platform for IoE devices with its inherent on-demand provisioning and elasticity for data storage, processing and UI, however a vast number of IoE use-cases are latency-sensitive and therefore long delays towards cloud providers are unsuitable. Medical applications often consist of devices with short transmission range and life-saving consequences [24][25] such as seizure detection [26]. Smart-cities provide a host of different applications which require low-latency and suffer from high device mobility, shared infrastructure/management [27] such as traffic management, public safety, [5] and emergency response. Industrial control Systems (as mentioned previously) not only require low-latency but may also operate in hostile environments [4], V2V etc. [27] [10].

This latency sensitivity has caused data processing to be pushed to the edge of the network, closer to its production. Fog computing is seen as combination of edge and cloud computing, providing the benefits of lower bandwidth and energy consumption whilst reducing greater QoS for IoE devices [27]. Fog is cited as a requirement for resilient IoE applications [28] Unfortunately fog nodes also suffer from fault-sensitivity due to their lack of scalable resources, typically wireless 1-hop device-to-fog communication and IoE device mobility [27]. To mitigate these issues, this work provides a highly resilient SaaS platform for fog networks, leveraging self-healing, self-organising and distributed processing capabilities.

III. PROPOSED MULTI-CELLULAR EMBRYONIC FOG PLATFORM

This section describes the proposed multi-cellular embryonic platform and its application to fog networks. This work extends a previous work in [29], where a conceptual model of an architecture based on the self-healing nature of embryonic cellular development was proposed as a solution to requiring a highly resilient architecture (figure 1) This architecture is purely distributed, having only a single component, the

cell. The cell self-reproduces as needed which allows it to scale with demand and also self-heal. Each cell differentiates according to a particular software function and complex applications can be composed through passing messages between different cells. The cells self-organise to provide a SaaS for data processing which is considered the most suited platform for IoE applications due to the devices' constrained nature [27] [30].

After further analysis of the networking characteristics of this architecture it was determined that too many communication aspects created a level of complexity which was not in line with the resilience requirements. To summarise, the ability to maintain any routing tables was incompatible with the high-levels of churn expected from environments which require high-resilience. Therefore a different, architectural approach was taken which involved local-only communication using multi-agent techniques inspired by embryonics. As a system for modelling discrete dynamical systems which operate using local-only communication, a Cellular Automata model was used to simulate the approach.

To summarise, this work investigates the application of embryonic techniques to a SaaS architecture. It aims to provide a combined Fog and IoE platform for highly resilient service delivery. Figure 2 illustrates the multicellular embryonic platform a top of the fog environment.

IV. CELLULAR AUTOMATA MODEL SIMULATION DESCRIPTION

This section describes the Cellular Automata (CA) based model for the embryonic architecture including the independent and dependent variables with motivations for their use. CA are discrete models of dynamical and complex systems. They are employed for modelling and simulating a variety of discrete scenarios. Within this simulation a stochastic CA is employed, due to having varying factors, such as the decision to spawn, the function to spawn and the simulated failure rate being dependent upon a random distribution. These stochastic cellular automata are sometimes referred to as Probability Cellular Automata (PCA) and these characteristics makes them suitable for simulation of the new model [31].

A. Definitions

The following definitions should be considered within the context of this model.

- **Cell** - the node of each network and the cell in the cellular automata. Also represents the biological cell within the model. It differentiates to a particular software function and will execute code of that function. Or it will be dead (state 0)
- **Function** - A logical software function e.g. cryptography, web service
- **Application** - Composed of multiple functions to provide a specific service. An application's functions must be able to reach each other in a consecutive manner.
- **Update Function** - Probabilistic function which determines the state of each cell within the simulation. Executed upon each cell. Updated Asynchronously and

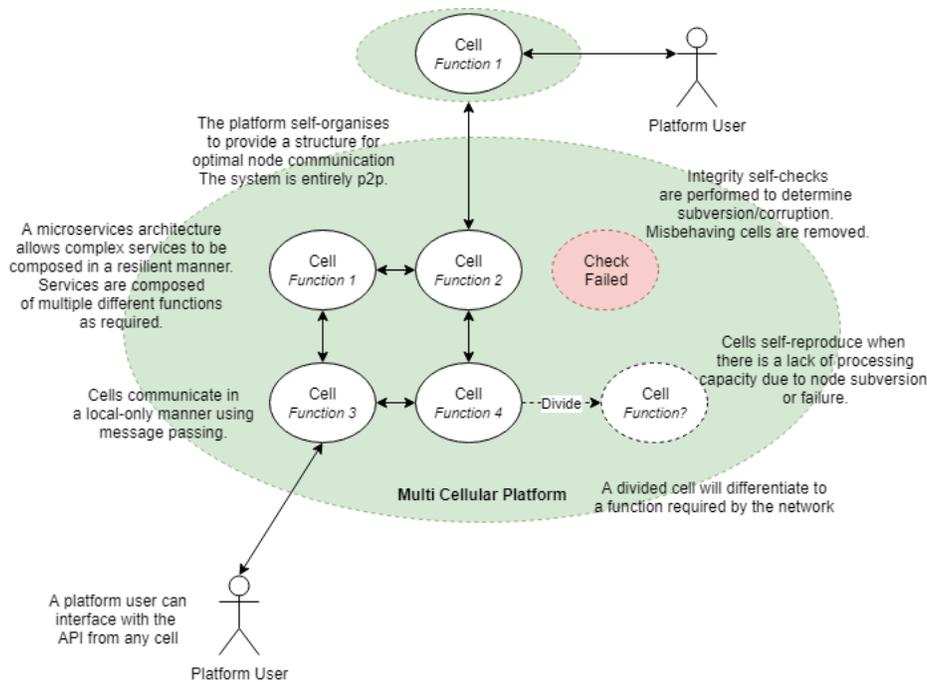


Fig. 1. The SaaS architecture inspired by animal embryonic development, composed of Multi-Cellular Platforms (MC). It provides resilient service delivery in hostile environments through leveraging the concepts of self-healing, cell division and differentiation.

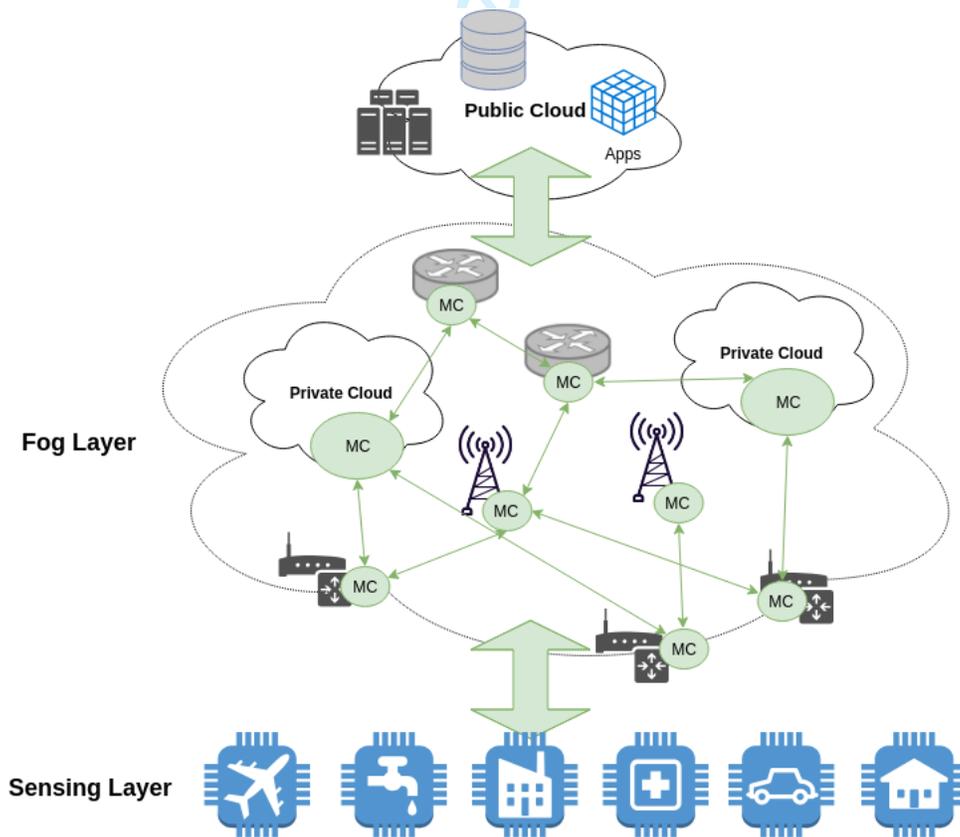


Fig. 2. The proposed Embryonic architecture situated within the Fog environment. MultiCellular platforms (MC) are distributed across the fog devices to provide data processing SaaS for the sensing layer of the SMART applications. This data is then offloaded to full cloud environments as required.

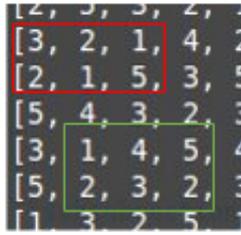


Fig. 3. The nodes in the red box are examples of a network which is not fully connected within the connected0 tests. The two starting nodes (state 1) can reach upto node 3 using local only communication but not to the final required function (state 5). In contrast the networking with a starting node highlighted in the green box can reach up to node 5.

randomly to account for similarities to real world networking.

- **Multi-Cellular Organism (MC)** - a collection of multiple cells which allows 1 or more applications to be executed via functions distributed across the cells.
- **Connectedness** - the quantity of fully connected applications. I.e. those where all functions can be reached consecutively in order to fully process the data.

B. Variables

The purpose of the simulation is to understand more about the constrained model's ability to resiliently deliver services. The external or internal changes to the system will be represented by the failure of nodes, according to a given stochastic variable. Whilst the resilience of the system will be measured through the number of services that are fully connected at any step within the simulation. This therefore will illustrate to what degree the network is still able to deliver its service under varying changes.

The following dependent variables (connectedness values) will be used to measure this resilience.

- **Connected0** - % of fully connected services with 0 networking hops allowed
- **Connected1** - % of fully connected services with 1 networking hop allowed

Connectedness is calculated as follows: at each step in the simulation, all starting nodes (state 1) are checked to see if they can reach the next consecutive function (state 2) and so on until the next function cannot be found, or if the final function is found creating a fully connected application. The quantity of fully connected networks is then recorded for that step in the simulation. Figure 3 illustrates two sub networks in a test network. One where the starting node can create a fully connected network and the other where it cannot.

With connected1, the search for each function is increased by one hop, as in figure 4. Neighbourhoods are von-neuman, where $r=1$ is a neighbourhood size of 5 inclusive of the central cell and $r=2$ is a neighbourhood size of 13, inclusive of the central cell. Von-neuman neighbourhoods were chosen over larger neighbourhood sizes, such as moore neighbourhoods, as it essential to reduce the number of communications being broadcast to nodes due to the flooding routing present in the network.

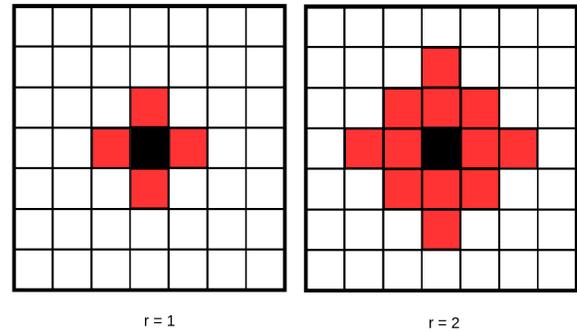


Fig. 4. Connected0 and Connected1 modelled as von-neuman neighbourhoods $r=1$ and $r=2$ respectively. The red squares indicate the nodes in the central squares neighbourhood, the nodes which will receive communications from the central, black node.

The following application related independent variables are used within the simulation to understand more about the self-healing characteristics:

- **Spawn Rate (SR)** - % chance a node will grow another cell
- **Death Rate (DR)** - % change a node will fail
- **Quantity of Functions (Q)** - the variety of different function types
- **Neighbourhood Size (N)** - quantity of adjacent nodes to each cell.

The following independent variables are employed to understand more about the complexity characteristics of the system and their effect upon the resilience of the network.

- **Starting Location** - the starting location of the cellular architecture in the grid. e.g. central, top-left, bottom-left etc.
- **Neighbour Start Size** - the initial size of the neighbourhood.

The initial cell update function is as follows: first a random variable will decree (according to a pre-set probability of failure) if the node fails. If not the node checks its local neighbourhood. If a node is found to contain the same function as the current node, then the current node will differentiate according to an under-represented function.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Time Steps	500
Grid Size	10x10
Nodes	100
Independent Variables	5
Runs per independent Variables	10
Total simulation runs	27000

V. SIMULATION MODEL

This section presents the simulation of the CA-based model presented in the previous section. Including implementation characteristics, validation and results. The model was implemented in pure python using test driven development and executed on an i5 Linux system with 4GB of ram. The grid

```

Step: 499

Connected nets: 8
[5, 22, 23, 17, 14, 19]
[2, 0, 3, 0, 3, 1, 5, 1, 3, 5]
[5, 2, 4, 1, 5, 3, 1, 2, 0, 4]
[2, 5, 3, 2, 1, 4, 3, 4, 1, 1]
[3, 2, 1, 4, 2, 2, 5, 1, 2, 4]
[2, 1, 5, 3, 5, 1, 3, 4, 0, 2]
[5, 4, 3, 2, 3, 2, 0, 1, 2, 4]
[3, 1, 4, 5, 4, 5, 1, 5, 1, 2]
[5, 2, 3, 2, 3, 4, 2, 4, 5, 1]
[1, 3, 2, 5, 1, 5, 1, 3, 2, 4]
[2, 1, 3, 1, 5, 2, 5, 2, 1, 5]

Averages:
[4, 18, 19, 19, 18, 19]
0.512479389909

```

Fig. 5. ASCII Graphical interface. The grid is composed of a number of cells which can be in states 0 to the number of functions. A fully connected network is one in which the functions can be reached consecutively.

of cells was represented as a 2D array, where the state of each cell is in the range 0 to function quantity. At each step in the simulation the cells were updated with the cell update algorithm.

A. Validation

1) *Graphical Validation*: The simulation model was validated through a number of different yet complimentary means.

Two graphical representations were used to validate the model simulation. The independent variables related to resilience could be adjusted prior to the simulation. The interface would then permit stepping through the simulation (up to the maximum 500 time steps) The graphical representations would then enable stepping through varying configurations. This was used to examine the system operating under varying levels of strain. For example, operating under a maximum spawn rate and zero death rate allowed easily validation that the software was representing the model correctly. Conversely, using a high failure rate and low spawn rate caused the system to be completely inactive. Both of these cases illustrate validation of the model through extreme condition testing.

2) *Extreme Condition Validation (Quantitative)*: Within the empirical experimentation dataset (presented in the next section) unstressed networks (zero failure rate) were tested. Verification occurred by examining only those tests where the $DR == 0$ and $SR == 1$, with the means of each quantity of function being approximately Grid size / Function quantity. Which illustrates that in a non hostile environment where the software is aggressively reproducing the random selection algorithm evenly distributes the function types. However as there is no cell destruction this network will converge to a solution with minimal dynamism and informs very little about the selection process effect upon resilience.

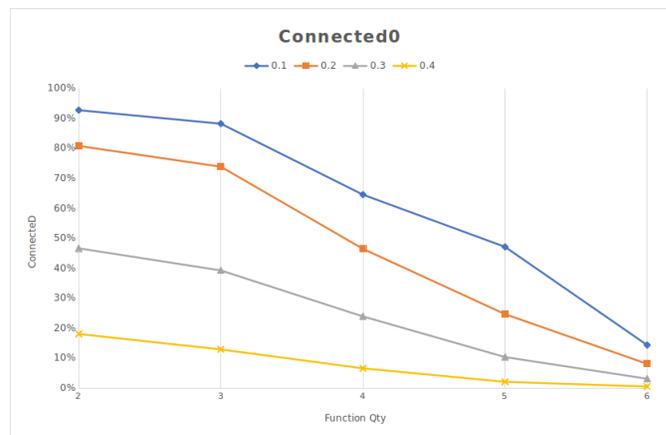


Fig. 6. Results of average connectedness of applications with 0 hop communication allowed.

3) *Internal Validity*: The empirical experimentation consisted of 27000 simulation test. These consisted of 10 runs of all independent variable permutations. The averages were then taken and due to the low variability and large amount of consistency (presented in the next section), these results seek to validate the simulation model.

B. Results and Analysis

The purpose of the simulation was to verify that the constrained system with no real routing could remain resilient i.e. continue to deliver services under varying network changes. As the services (applications) will successfully execute when all functions of each application are fully connected, the goal is to examine the quantity of connected applications at each time step. This connectedness of each network is then compared against the independent variables (application and complexity characteristics) mentioned in the previous section.

The groups of tests examined in this section are simulated for a MC under stress, where $DR > 0$. We are seeking to determine how well the applications can still communicate using local only-interaction when under varying levels of network stress (death rate) and application complexity (function types 2-6). Variables such as spawnrate and cell differentiation process were examined in order to optimise the performance of the MC.

1) *SpawnRate*: During an initial analysis, it was shown that through aggressive spawning a performance improvement could be made. Conversely, reducing the spawnrate always caused negative performance. Figures 6 and 7 show the connected0 tests where $SR == All$ and $SR == 1$, respectfully. The smoother declines indicates that this small optimisation can increase the ability for the applications to be connected, where higher failure rates benefit more over lower failure rates. This is due to the aggressive spawning increases the likelihood of nodes filling gaps where previous nodes had failed (state 0) i.e there is a greater number of functional nodes and a reduced number of failed nodes. At lower failure rates this has less of an effect due to lower quantity of failed nodes.

2) *Cell Update Function*: Another method of performance optimisation made within the CA is the update function, which

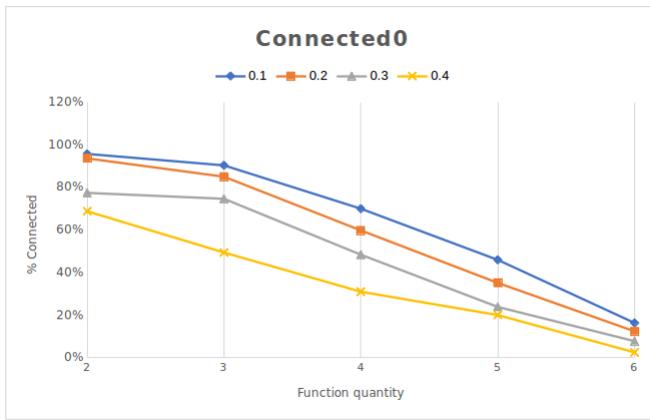


Fig. 7. Results of average connectedness of applications with 0 hop communication allowed where spawnrate==1

may choose to differentiate cells in order to optimise the MC. Therefore this section presents the results of 2 different groups of tests, each with different update functions. The update functions are as follows:

- **No Differentiation Optimisation (NoDif)** - Cells do not attempt to optimise the distribution of cells except to remove redundancy. The cell firstly examines the states of all adjacent cells. If it's state is the same as an adjacent cell and the quantity of functions is greater than the neighbourhood size, it will attempt to differentiate to to an unseen state. The cell will still divide to a random function when there is adjacent space (Algorithm 1).
- **Differentiate according to weighting (Dif2)** - As above the cell first examines the adjacent node states and determines the least represented functions out of the global set of functions. It will then differentiate to the least represented function or if there are many of the same weighting, a random function from this subset (Algorithm 3).

Figure 8 presents the results of network connectedness for the connected0 tests. A clear trend can be seen that as the quantity of functions increases the ability for the applications to remain connected decreases. This is caused by the complexity of the application ensuring the likelihood of one function being adjacent to another to reduce. In the currently measured von-neumann neighbourhood where $r=1$, a cell will be able to communicate with 4 other cells. Therefore if $Q > 4$ the chance of finding the next node is < 100 . This explains the considerable performance drop at functions 4 or greater. Regarding the difference between NoDif and Dif2, the trend illustrates dif2 providing increased connectivity over NoDif where $Q \geq N$. This performance increase appears to be strong at medium DR of 0.2 and 0.3 but less favourable at 0.4. This evidences that attempting to optimise the distribution of cells does increase performance but only to a certain failure rate.

Figure 9 presents the results of network connectedness for the connected1 tests. The same trend can be seen as in connected0, however this time the decay is slower and the

Algorithm 1: Cell Update - No differentiation Optimisation

Data: NeighbourhoodState, CellState, Functions

Result: CellState, DivideCellState

Check Neighbourhood States;

if Cell is alive **then**

if Cell does not die **then**

if CellState is in NeighbourhoodState and $N > 4$ **then**

 CellState = random from Functions not in NeighbourhoodState;

end

if Cell should divide **then**

 Check Functions in NeighbourhoodState;

for Function in Functions **do**

if Function not in NeighbourhoodState **then**

 DivideCellState = Function;

else

 DivideCellState = random from Functions;

end

end

 Divide with DivideCellState;

end

else

 CellState = dead;

end

end

decline is a lot smoother, there are no sudden jumps (such as when $Q > N$). This would suggest that the issue of reduced probability is mitigated as the size of the search space is now increased. If it is assumed that the self-organisation of the MC will force differentiation of under-represented cells within the same time scale as the communication to next node, then the search space at each step will be $N * N$, else it will be $N * (N - 1)$. A considerable increase over simply N . The function distribution algorithm dif2 follows a similar performance improvement trend as with connected0 and therefore highlights the efficacy of this approach. As with the connected0 tests, high failure rates and tests where $Q > N$ still have poor performance.

C. Organism Start Size and Location

Starting location was examined for its relationship to complex systems, in case it had an effect upon the resilience of the MC. However due to a correlation of < 0.009 it was determined to have no effect.

A characteristic of the MC which is internally adjustable and is also concerned with complex systems is its starting size, i.e. the number of starting cells. Test groups were run with starting cells of 1, 5 and 9. For all tests, as start size increases, as does the connectedness. Which is an intuitive finding. At high failure rates, this can be quite a considerable performance increase of 50%, where $Q < N$.

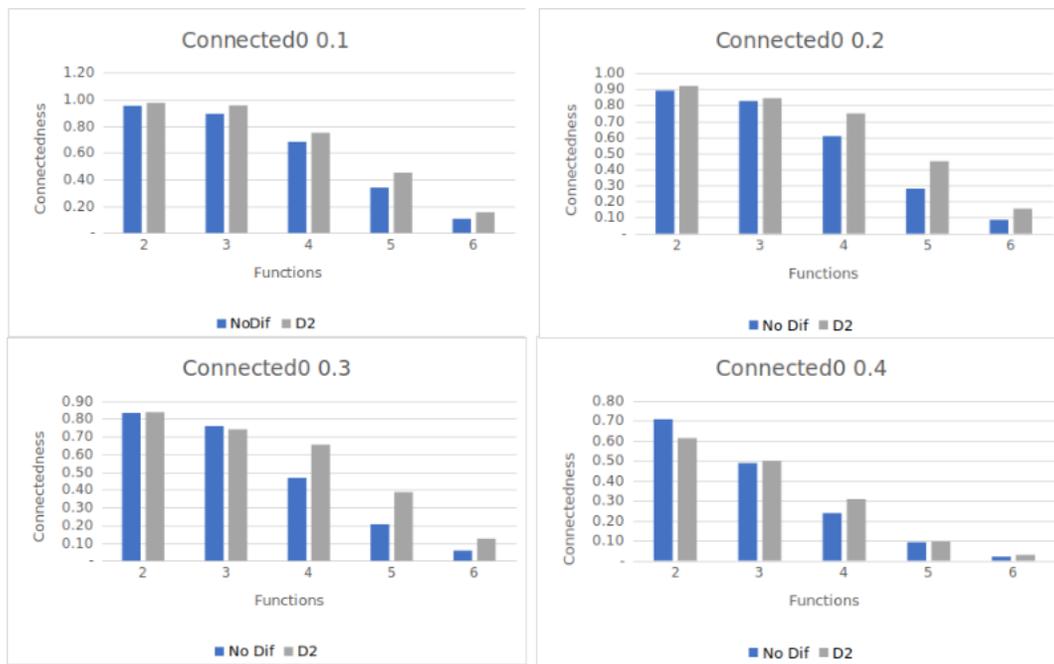


Fig. 8. Average connectedness comparing differentiation methods for connected0 tests

Algorithm 2: Cell Update - Weighted Differentiation

Data: NeighbourhoodState, CellState, Functions

Result: CellState, DivideCellState

Check Neighbourhood States;

if Cell is alive **then**

if Cell does not die **then**

for Function in Functions **do**

 Count Function in NeighbourhoodState;

end

 minFunctions = Calculate min(Function in NeighbourhoodState);

 CellState = random(minFunctions)

if Cell should divide **then**

 Check Functions in NeighbourhoodState;

for Function in Functions **do**

if Function not in NeighbourhoodState

then

 DivideCellState = Function;

else

 DivideCellState = random from Functions;

end

end

 Divide with DivideCellState;

end

else

 CellState = dead;

end

end

Figures 10 and 11 show the distribution of tests which ended up with either an early failure or successfully survived, with the corresponding connectedness. In figure 10, the results for a starting MC of 1 show that a greater number of tests would fail than succeed. However figure 11 shows the same distribution but for a starting neighbourhood of 9 where only a small number of tests failed. This illustrates the effect of starting size upon success.

These graphs highlight some interesting aspects relating to the efficacy of different intelligent dynamic differentiation algorithms. At the higher start sizes the algorithms appear to be almost on par. However where start size == 1, the results don't fit a clear trend. Closer examination of these groups of tests indicates something interesting. That the test results were polarised where they either tend to be highly connected, ($> 0.9\%$) or not at all which explains the poor performance where $DR = 0.3$ in Dif1. This is a trend that is persistent throughout the tests where those that did not fail had consistent connectedness where the performance improvement could be gained by increasing the start size.

Examining this at the highest failure rate of 0.4 illustrates still good performance, however this only holds true where $N > 4$. This strongly illustrates that the initial stages of an MC are vital to its ongoing survival.

VI. ANALYSIS

This section presents an analysis of the results provided in the previous section. The results of all tests have indicated a number of points concerning factors which affect the connectedness of applications, within the embryonic model. All of which highlight methods to optimise the resilience of the MC dependent upon its characteristics. The independent factors (those variables which can be internally adjustable by

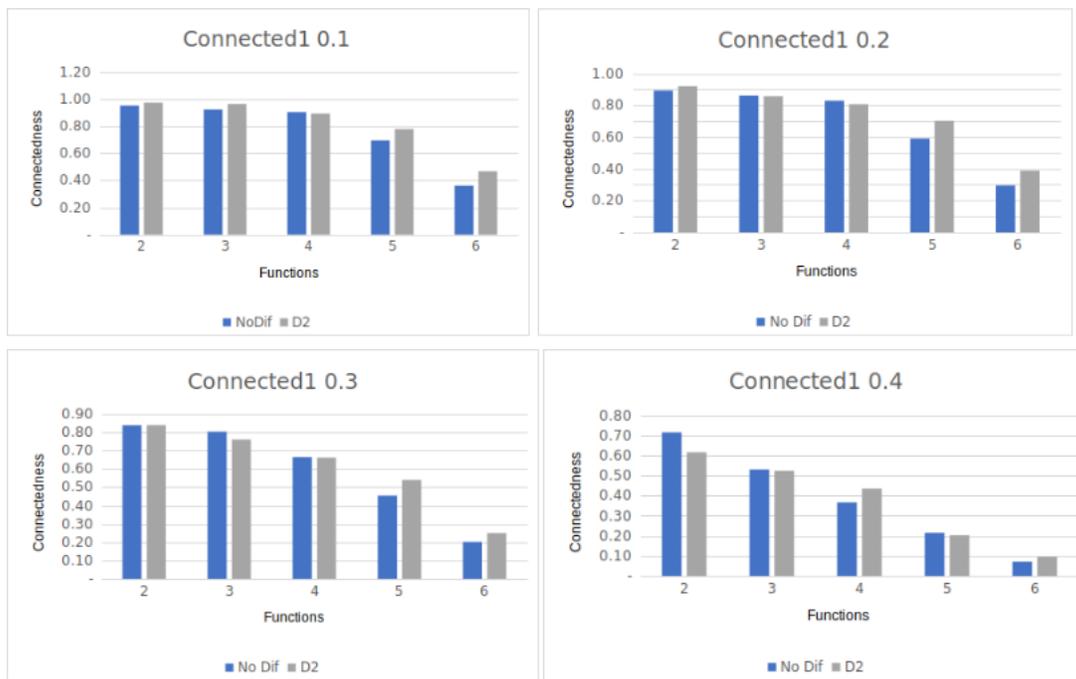


Fig. 9. Average connectedness comparing differentiation methods for connected1 tests

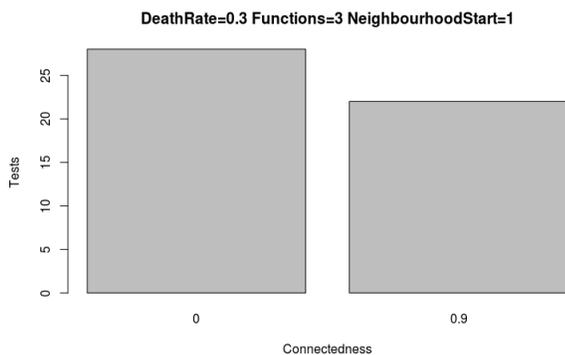


Fig. 10. 50 test runs with death rate 0.3, functions 3 and Neighbourhood startsize 1. Showing just under 50% of tests were successful whilst the rest were unsuccessful.

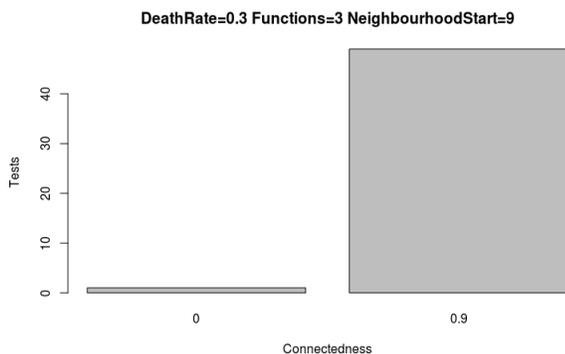


Fig. 11. 50 test runs with deathrate 0.3, functions 3 and Neighbourhood startsize 9. Where the majority of tests were highly successful.

the MC) are: the application complexity (quantity of functions per application), the level of division aggressiveness, starting MC size and the algorithm to determine differentiation. All of which are discussed in further detail below.

A. Application Complexity

The quantity of functions within an application undoubtedly has a direct effect upon its connectedness. This is by the inherent nature of needing to connect more cells within an application, leaving more points for failure. However, dividing application functionality is a prime characteristic of the MC architecture and a requirement of resilience, in order to distribute risk. Therefore any technique which increases connectivity under increased application complexity is a positive benefit to the resilience of the embryonic architecture. A clear point is that adjusting the complexity of an application can enable it to survive during periods of particular stress although at an obvious reduction of functionality. This can pave the way for networking algorithms that downgrade according to perceived performance.

If it is not feasible to manage complexity through reduction of functionality, applications could be divided into sub sections or "organs" following the biological model. This would cause the overall likelihood of the application to still execute successfully albeit with a largely increased resource usage.

Whilst the application complexity would be reduced, communication protocols and network management complexity would increase due to the additional network overlay. Therefore finding an appropriate balance would be necessary.

B. Aggressive Spawning

In the previous experiments it was quickly determined that given the option, dividing to create a new cell increased

the connectedness within the MC. The performance increase was considerable at higher failure rates but also decreased proportionally to application complexity. This is intuitive as a greater number of cells permits a higher chance of finding the next required function. This is directly through the required function being adjacent to the cell and indirectly through enhanced communication. However, during practical testing, the time period for spawning will need to be determined appropriately so as to not delay execution of other components with the cell, e.g. function execution.

C. Complex System Characteristics

Two characteristics of complex systems were also investigated for their effect upon connectedness, the cell's initial starting size and location. The start location showed little benefit as varying it made no difference to the connectedness results with a almost zero correlation. However the starting size correlated with an increase in connectedness, particularly at higher failure rates. The reason for this is the same as the aggressive spawning point previously mentioned. This is a strong finding considering this could give networks starting in highly hostile conditions an increased chance of survival. As it was noted that networks either survived entirely or failed early on, this highlighted the necessity for ensuring strong survival in early stages, where increased MC start size is a strong enabling factor.

D. Increasing Search Space

Throughout all the results, a clear improvement can be seen with the connected1 tests over the connected0. As explained previously, this is due to increasing the potential search space through the increased neighbourhood size, which can also be understood as von-neuman networks where $r=1$ and $r=2$, respectfully. If the quantity of functions is greater than the neighbourhood size (which is decreased further by the failure rate) then the probability of connectedness decreases and therefore increasing this search space can mitigate this issue and thus improve the connectedness of the service. However due to the application of flooding based routing, larger sized neighbourhoods will have a detrimental affect upon MC performance. Through permitting 1 hop communication, the search space can be considerably increased and thus the likelihood of remaining connected increases. Optimisations will need be determined once more is known about the effect of communication upon the MC in later stages of the research.

Therefore, we can deduce from the previous analysis that the relationship between the neighbourhood size and function quantity (service complexity) has a direct effect upon the connectedness of that application. The probabilistic model presented in section 4.1 indicates that during a best case scenario, the most optimal selection for function quantity is any which is less than the neighbourhood size. This can be explained as follows:

If $Q > N$ then the chance of the application to remaining connected is $< 100\%$, which reduces considerably with larger service sizes. Therefore increasing the neighbourhood size, and in particular ensuring it is larger than the application

size including failure rate DR , such that is $N > Q$ or $N > (Q * DR)$ will cause a considerably increased chance for the application to remain connected.

E. Function spread through differentiation algorithms

Another point of optimisation is the method used by cells to differentiate according to the needed functions. The probabilistic model illustrates a best case scenario where function types are evenly distributed. However, as a result of node failures and possible differentiation methods the CA is incredibly dynamic, causing cell state to be in a constant state of flux, potentially changing at each time step. Therefore through deriving intelligent mechanisms which differentiate to force a given spread of functions, the connectedness can be improved. This can be seen with the performance improvement between the two differentiation algorithms, particularly where $Q > N$ and at higher failure rates. However this performance does not reach that of the probabilistic model although it does help to confirm its findings. This is largely due to the highly dynamic nature of the CA which also lacks many features making it less representative of the real-world system. The next stage of this research will develop a prototype implementation to permit investigation of the model with the currently lacking communication and processing performance attributes.

VII. CONCLUSION

In the authors' previous work [29], modelling of the embryonic characteristics against the required functionality highlighted a disparity between goals. Whilst the networking functionality decreed an array of features such as routing tables; the resilience and complexity characteristics were at odds, due to the increase in attack surface. Therefore this paper provided models and corresponding simulations to understand the efficacy of the embryonic model yet with the negative networking aspects removed: i.e. local-only communication. This enabled an investigation into the architecture structure of the model without skewing the results with complexity of the communication.

Overall the results of the CA simulations have highlighted a number of key points relevant to the next (practical) stage of the investigation: Structural characteristics can be varied in order to increase resilience under varying levels of stress. These will provide the baseline categories for the next stages of tests. Increasing structural characteristics tend to be resource intensive in all manners, so their usefulness in a practical context will be quantified when using real-world systems. Further methods of intelligent decision making relating to optimising the function spread when choosing to differentiate can be investigated in a greater manner, as the discrete nature of the CA simulation prevented this but did highlight its effectiveness.

REFERENCES

- [1] J. P. G. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Comput. Netw.*, vol. 54, no. 8, pp. 1245–1265, Jun. 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.comnet.2010.03.005>

- 1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
- [2] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz, "Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges," *IEEE Communications Surveys Tutorials*, vol. 14, no. 2, pp. 607–640, Second 2012.
- [3] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial internet of things and industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4674–4682, Oct 2018.
- [4] A. Singh, U. K. Singh, and D. Kumar, "Iot in mining for sensing, monitoring and prediction of underground mines roof support," in *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, March 2018, pp. 1–5.
- [5] A. J. V. Neto, Z. Zhao, J. J. P. C. Rodrigues, H. B. Camboim, and T. Braun, "Fog-based crime-assistance in smart iot transportation system," *IEEE Access*, vol. 6, pp. 11 101–11 111, 2018.
- [6] S. Heble, A. Kumar, K. V. V. D. Prasad, S. Samirana, P. Rajalakshmi, and U. B. Desai, "A low power iot network for smart agriculture," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, Feb 2018, pp. 609–614.
- [7] R. Amin, D. Ripplinger, D. Mehta, and B. Cheng, "Design considerations in applying disruption tolerant networking to tactical edge networks," *IEEE Communications Magazine*, vol. 53, no. 10, pp. 32–38, October 2015.
- [8] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of threats to the internet of things," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018.
- [9] T. Welsh and E. Benkhelifa, "Perspectives on resilience in cloud computing: Review and trends," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, Oct 2017, pp. 696–703.
- [10] E. Baccarelli, P. G. V. Naranjo, M. Scarpiniti, M. Shojafar, and J. H. Abawajy, "Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study," *IEEE Access*, vol. 5, pp. 9882–9910, 2017.
- [11] M. A. A. Faruque and K. Vatanparvar, "Energy management-as-a-service over fog computing platform," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 161–169, April 2016.
- [12] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "Ppfa: Privacy preserving fog-enabled aggregation in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3733–3744, Aug 2018.
- [13] F. Dressler and O. B. Akan, "A survey on bio-inspired networking," *Computer Networks*, vol. 54, no. 6, pp. 881 – 900, 2010, new Network Paradigms. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1389128610000241>
- [14] S. N. Mthunzi, E. BENKHELIFA, T. Bosakowski, and S. Hariri, "A bio-inspired approach to cyber security," in *Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives*, November 2018. [Online]. Available: <http://eprints.staffs.ac.uk/5068/>
- [15] S. Hariri, M. Eltoweissy, and Y. Al-Nashif, "Biorac: Biologically inspired resilient autonomic cloud," in *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research*, ser. CSIIRW '11. New York, NY, USA: ACM, 2011, pp. 80:1–80:1. [Online]. Available: <http://doi.acm.org/10.1145/2179298.2179389>
- [16] B. Baran and R. Sosa, "A new approach for antnet routing," in *Proceedings Ninth International Conference on Computer Communications and Networks (Cat.No.00EX440)*, Oct 2000, pp. 303–308.
- [17] D. Wang, L. He, Y. Xue, and Y. Dong, "Exploiting artificial immune systems to detect unknown dos attacks in real-time," in *2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, vol. 02, Oct 2012, pp. 646–650.
- [18] W. Ha, "Cloud service selection with fuzzy c-means artificial immune network memory classifier," in *2018 14th International Conference on Computational Intelligence and Security (CIS)*, Nov 2018, pp. 264–268.
- [19] T. Wang, B. Ye, Y. Li, and Y. Yang, "Family gene based cloud trust model," in *2010 International Conference on Educational and Network Technology*. IEEE, 2010, pp. 540–544.
- [20] P. Marchal, P. Nussbaum, C. Pignet, S. Durand, D. Mange, E. Sanchez, A. Stauffer, and G. Tempesti, "Embryonics: The birth of synthetic life," in *Towards Evolvable Hardware*, E. Sanchez and M. Tomassini, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 166–196.
- [21] E. Benkhelifa, A. Pipe, and A. Tiwari, "Evolvable embryonics: 2-in-1 approach to self-healing systems," *Procedia CIRP*, vol. 11, pp. 394 – 399, 2013, 2nd International Through-life Engineering Services Conference. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2212827113005027>
- [22] D. Miorandi, D. Lowe, and L. Yamamoto, "Embryonic models for self-healing distributed services," in *Bioinspired Models of Network, Information, and Computing Systems*, E. Altman, I. Carrera, R. El-Azouzi, E. Hart, and Y. Hayel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 152–166.
- [23] K. Velasquez, D. P. Abreu, M. R. M. Assis, C. Senna, D. F. Aranha, L. F. Bittencourt, N. Laranjeiro, M. Curado, M. Vieira, E. Monteiro, and E. Madeira, "Fog orchestration for the internet of everything: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 9, no. 1, p. 14, Jul 2018. [Online]. Available: <https://doi.org/10.1186/s13174-018-0086-3>
- [24] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, "Towards fog-driven iot ehealth: Promises and challenges of iot in medicine and healthcare," *Future Generation Computer Systems*, vol. 78, pp. 659–676, 2018.
- [25] F. A. Kraemer, A. E. Braten, N. Tamkittikhun, and D. Palma, "Fog computing in healthcare: a review and discussion," *IEEE Access*, vol. 5, pp. 9206–9222, 2017.
- [26] M.-P. Hosseini, A. Hajisami, and D. Pompili, "Real-time epileptic seizure detection from eeg signals via random subspace ensemble learning," in *Autonomic Computing (ICAC), 2016 IEEE International Conference on*. IEEE, 2016, pp. 209–218.
- [27] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," *Computer*, vol. 49, pp. 112–116, 2016.
- [28] C. C. Byers and P. Wetterwald, "Fog computing distributing data and intelligence for resiliency and scale necessary for iot: The internet of things (ubiquity symposium)," *Ubiquity*, vol. 2015, no. November, pp. 4:1–4:12, Nov. 2015. [Online]. Available: <http://doi.acm.org/10.1145/2822875>
- [29] T. Welsh and E. Benkhelifa, "Embyronic model for highly resilient paas," in *2018 Fifth International Conference on Software Defined Systems (SDS)*, April 2018, pp. 197–204.
- [30] S. V. Vandebroek, "1.2 three pillars enabling the internet of everything: Smart everyday objects, information-centric networks, and automated real-time insights," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 14–20.
- [31] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan 2003.