Interacting Hierarchical Dynamic Networks

Peter Meltzer[1,2] and Peter J. Bentley[1,2]

[1]University College London, London, UK
[2]Braintree Ltd, Gower Street, London UK
p.meltzer@cs.ucl.ac.uk

Abstract

In this work we present IHDNs: an original model of computation for the simulation of interacting, dynamic, multi-scale systems. We show that a novel message passing mechanism that operates across layers of abstraction in hierarchical dynamic networks is effective in expressing the complex dependencies of living systems. Using a conventional computational model of cell evolution in cancerous tumour growth for comparison, we demonstrate the validity of IHDNs in emulating the behaviour of life-like systems, as well as the additional capabilities in enabling Neo4j Cypher pattern-matching queries, demonstrated here in the analysis of evolutionary cell heritage.

Introduction

Hierarchy is overwhelmingly evident in every aspect of life, emerging in any imaginable circumstance as a direct consequence of evolution. Simple structural hierarchies can be seen in everything from the organisation of the stars to object-oriented programs. However, in biological contexts, the hierarchies that emerge are often *dynamic*, and involve complex dependencies between components that do not exist at the same scales.

A typical example of dynamic hierarchy is that of proteins, cells, and biological organisms. In this case, the building blocks (proteins) have a direct impact on the behaviour of the higher order entities (cells and higher still, organisms), but higher order functions equally have an impact on the configuration of the building blocks (Lenaerts et al., 2002).

Moreover, interactions between the lower order entities may also affect the entity indirectly, and its composition may change. For example, it is the interaction of particular proteins that facilitate the protein aggregation mechanism by which the neuronal degeneration of Huntingdon's disease is caused (Gonzalez and Kann, 2012). These interactions within and between biological hierarchical networks make them highly dynamic, potentially changing every part of themselves from the organisation to the functioning of the components.

Due to their abundance in nature, the utility in simulating complex dynamic networks cannot be overstated; with modelling applications in medicine, biology, macro-economics, and other ecological sciences. However, capturing the complexities of interacting multi-scale systems that are able to change their internal configurations and behaviours dynamically in a computational model is not a trivial task.

In this paper we propose an original, graph-based model of computation for the simulation of Interacting Hierarchical Dynamic Networks (IHDNs), where the representation of components of different scales combined with a novel cross-layer message passing system enables the simulation of complex adaptive systems across any scales of abstraction.

In order to present the proposed model, first we review relevant literature. Then after presenting the concepts and architecture of the model, we demonstrate its ability to emulate the behaviour of living systems with the simulation and analysis of tumour growth in a dynamic evolving cell network. We verify our results against an existing model of this phenomenon (Araujo, 2013), that has also been used to test another unconventional model of computation (Sakellariou and Bentley, 2015), then we show the advantages of the model for deeper analysis. Finally, we present our conclusions.

Source code for both the computational platform and the demonstrated aneuploid tumour growth simulation are available for reference at `github.com/meltzerpete/ihdn`.

Background

The modelling of complex dynamic systems has employed the use of many solutions; including Agent-Based Models (ABMs) (Gilbert, 2004), and more recently Dynamic Networks (Sayama, 2007). In this section we review a sample of these solutions.

Although emerging from the object-oriented programming paradigm, ABMs have many parallels with Cellular Automata (CAs), which have also been used to model complex systems (Batty et al., 1999; Webster and Malcolm, 2008). ABMs have been particularly popular in modelling complex social systems in order to observe emergent and

collective behaviours (Axelrod, 1997).

The modelling methodology for ABMs typically begins with (deductive) observations of real world phenomena in order to derive agent state update rules. With the agents' update rules defined, simulations can be executed wherein (inductive) analysis of emergent properties can be made. Consequently, Axelrod (1997) and Epstein (1999) argue that ABMs offer a distinct "third" scientific method i.e. *generative* science.

In more recent years, Multi-Agent Systems have been developed (Ahmed et al., 2006), and ABMs have been combined with Reinforcement Learning (Arel et al., 2010), in which agents' policy functions are optimised to minimize the distance between simulated and real-world observed data. In an attempt to better capture the inherent hierarchy in naturally occurring complex systems, models such as (Bortot et al., 2017) and (Yao and Van de Peer, 2017) define layers for hierarchical organisation of agents. However, these models only allow for a finite number of layers and configurations; and hence, as with ABMs in general, are restricted in their representation of dynamic systems.

Different in their approach, Dynamic Networks have been applied to modelling complex phenomena found in epidemics (Gross et al., 2006), social networks (Funk et al., 2010), and neuroscience (Pearlmutter and Houghton, 2009). However, these applications are typically concerned with either the changing states of fixed topology networks (of which conventional Artificial Neural Networks are a prime example), or the changing properties of a network based on topological transformations alone.

Contrary to this, (Sayama, 2007) provides a framework for the uniform representation of state-topology co-evolution via graph-rewritings, with a demonstration of automated rule discovery using real-world observed network evolution data (Sayama et al., 2013). However, as a consequence of decoupling the representation of entities and their behaviours, these models do not achieve the same expressiveness of ABMs in describing the effects of small changes in individual systems on the dynamics of the whole (Bentley, 2009).

Interacting Hierarchical Dynamic Networks

The Model of Computation

The single component of computation in IHDNs is the system (Figure 1). As in a property graph, a system S may have a set of any number of labels L^S, and a set of any number of properties P^S in the form of key value pairs. In addition, a system may optionally be given a vote function F_V^S, a filter vector I^S, and a vote vector V^S. An ordered set of all possible system functions F is shared by all systems, with $|F|$ denoting the total number of defined functions.

The system may have a set of any number of child systems $C^S = \{C_1^S, C_2^S, ...\}$ and (excluding `ROOT` systems) will have always at least one parent system P^S. Relationships indicating compositional hierarchy are labelled with the `CONTAINS` relationship type, while any other user defined relationship types may exist between any pair of systems. Typical operations for topology mutation include `deepClone` and `transfer` (Figure 2).

Computation (here the update of system state - internal

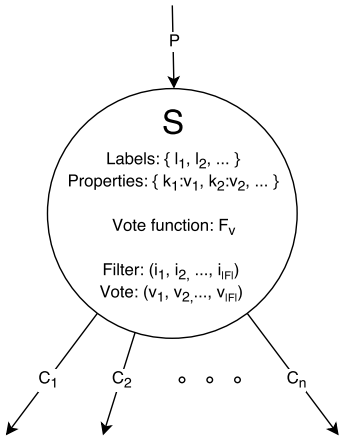

Figure 1: The abstract IHDN system model.

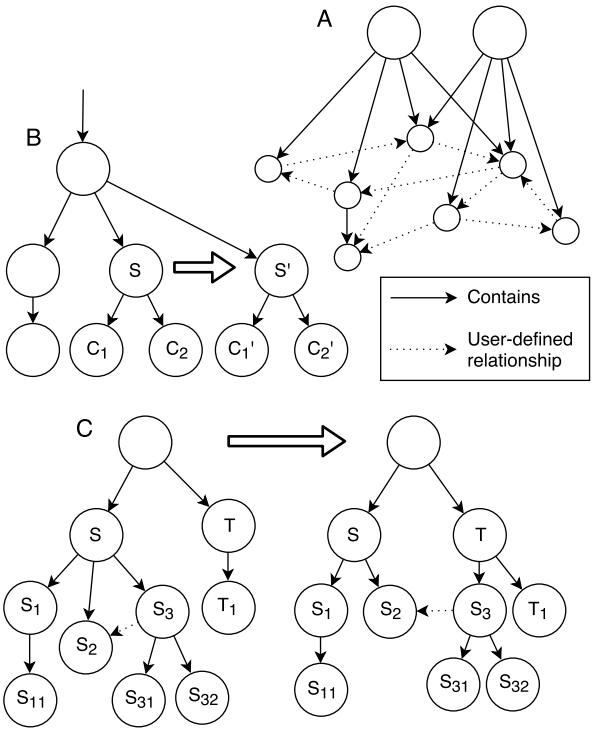

Figure 2: (A) Systems may have multiple parent systems, and user defined relationships may exist between systems of the same or different scales. (B) A `deepClone` operation on system S recursively copies contained systems, while membership in higher order entities is inherited. (C) The system S performs a `transfer` operation on S_3 to the system T.

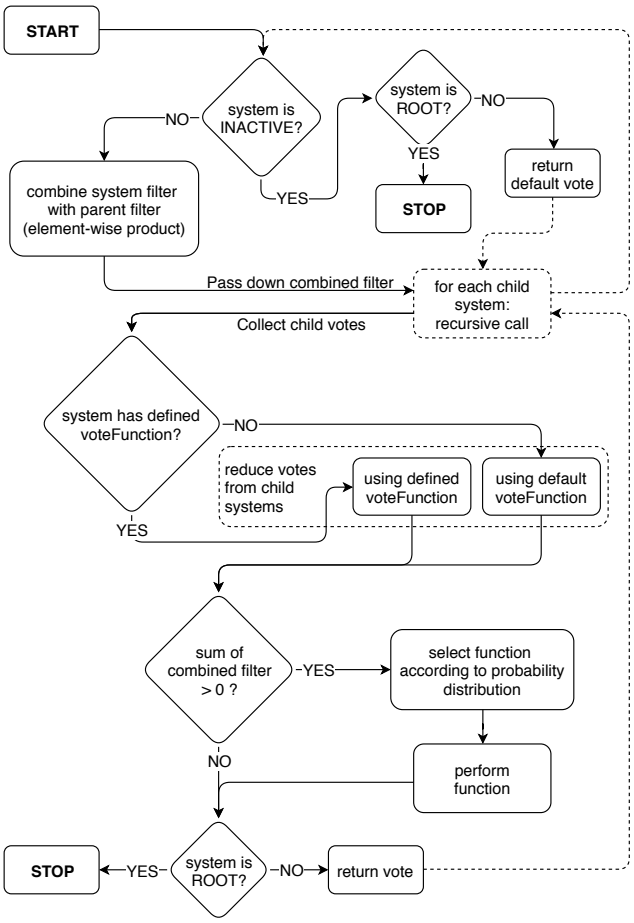

Figure 3: Recursive compute algorithm executed at each ROOT system once per iteration.

and/or structural), then proceeds via a depth-first traversal over the CONTAINS relationships initiated at every ROOT system (in a random order), once per *iteration*. The traversal facilitates a message passing system that enables systems to influence the selected actions of others at different levels in the hierarchy.

When each system is selected to perform an action, the functions are selected from F probabilistically, according to the messages received by that system. There are two types of message passed between systems in the hierarchy: filters are passed down the tree; and votes are passed upwards. The elements of both the filter and vote vectors correspond directly to the functions of F.

To select a function for a system S to perform, the filter vector from the current parent system $I^P$and the set of vote vectors V^{C^S} are both considered. As filters are passed down the tree, they are combined with the element-wise product, enabling a system to set any chosen function's probability of selection to 0 regardless of the received votes and filters of lower level systems. Equally filters may introduce an overall bias to be applied to the received votes and to hierarchically bias the actions of lower constituent systems. The default behaviour is to reduce V^{C^S} with addition to give

$$V^S = V^{C_1^S} + V^{C_2^S} + \cdots + V^{C_n^S}$$

to then calculate the element-wise product $V^S \odot I^P$. The result is a vector of length $|F|$, which is used as a probability distribution relative to the sum of its elements to select with bias the function $f \in F$ to perform.

While the default behaviour is to combine child systems' votes with addition, and adding the system's own vote before passing the vote up the tree, the votes may be intercepted and reduced differently, or even completely discarded by defining a new vote function for any chosen types of system.

Figure 3 provides an outline of the recursive compute function, and demonstrates the order in which filters and votes are combined and how they are passed up and down the tree.

Implementation

The IHDN prototype implementation used for demonstration here is written in Java and exposes a simple API for the development of simulations on top of an embedded Neo4j instance. If votes and filters are not specified for a given system, the defaults of $(0, 0, \dots)$ and $(1, 1, \dots)$ are used respectively, contributing no bias or restriction on function selection. Likewise, any ROOT systems are given the default filter of $(1, 1, \dots)$ as their parent filter during computation.

On completion of a simulation, the graph database is stored and can then be queried directly with Cypher[1] , or visually using any existing Neo4j compatible tools. The demonstration that follows employs the use of the Neo4j multi-platform desktop browser.

Tracing Cell Lineage in Simulated Aneuploid Tumour Growth

Having presented the IHDN platform for the simulation of dynamic hierarchical systems, we now evaluate it against an existing biological model of cancerous tumour growth implemented on a conventional computer (Araujo, 2013), which was also used to test another unconventional computing platform (Sakellariou and Bentley, 2015). The rest of this section describes the simulation according to the methodology given in (Bentley, 2009).

The simulation is concerned with role of chromosome missegregation in cancerous tumour growth; and since cancer is progressive via heritable change to cells, the relevance of tracing this change in understanding and hence treating cancer is especially evident. Better understanding of cell lineage affords greater understanding of how a particular cancer will progress, how susceptible it will be to treatment, and the likely-hood of its return (Frank, 2007; Bolton et al., 2016).

[1] https://neo4j.com/cypher-graph-query-language/

Biological Observations

During normal mitotic cell division, each chromosome is duplicated and the resulting set of chromosomes is segregated equally (in a direct one to one correspondence) between the resulting new cells. However, it is estimated for human cells that an average of one in one hundred cell divisions spontaneously missegregates (Cremer et al., 2001), i.e. fails to separate the chromosomes into two identical sets, resulting in one cell with extra chromosomes (and hence extra copies of the contained genes), and the other cell with fewer.

As a consequence of this phenomenon and the configuration of which genes are present in the gained and lost chromosomes, cells with different properties and behaviours arise, which can lead to the evolution of cancerous cells that divide highly and do not die naturally.

The Model

To explain the way in which aneuploid tumours develop requires four main abstractions of physical systems: the tissue, the cell, the chromosome, and the gene. Contrary to previous implementations, here these abstractions are adopted directly (each as a tree of systems in the hierarchy), affording an intuitive correspondence between the problem domain and the computational model.

Components

For this particular simulation, it is necessary to model three particular gene abstractions - the apoptosis regulatory gene (a tumour suppressor gene that regulates cell death), the cell division regulatory gene (an abstraction of proto-oncogenes that promotes cell growth and progression), and the chromosome segregation regulatory gene (an abstraction of genes that control the fidelity of cell division and reduce the likelihood of chromosome missegregation). To capture the sensitivity of the cell behaviours on the initial genetic configurations and gene linkage (the membership of which genes are encoded in which chromosomes), three different chromosome distributions are modelled (Figure 4).

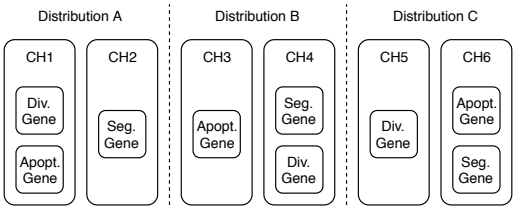

Figure 4: Three possible chromosome distributions, formed as combinations of six unqiue chromsome configurations.

A complete list of the entities modelled in this experiment can be seen in Table 1. As will be discussed below, each physical entity is not represented by a single system, but rather the composition of a hierarchy of systems. Thus each physical entity (i.e. the tissue, cell, etc.) is actually represented by a tree of systems, with the corresponding system as its root.

To simplify any analytical computation, the possible chromosome configurations are labelled with `CH1` to `CH6` according to the six different possible combinations (Figure 4). Additionally, upon completion of the simulation each `CELL` system is given a `genome` property in the form of a vector representing the number of each type of gene that it contains. However, neither the additional labels or property are required during the computation.

Table 1: The set of all IHDN system types used in this simulation.[2]

IHDN System	Notes
Tissue	Labels: `TISSUE`, `ROOT`
Cell	Labels: `CELL` Properties: `start`, `nDivs` Vote function: `cellVote`
Dead cell	Labels: `CELL`, `INACTIVE` Properties: `start`, `nDivs`, `inactiveAt`
Cell copy	Labels: `CELL_COPY`, `INACTIVE` Properties: `start`, `nDivs`, `missegregationAt`
Chromosome	Labels: `CHROMOSOME`, one of $\{$`CH1`, ..., `CH6`$\}$ Filter: $(0,0,0)$
Apoptosis gene	Labels: `GENE`, `APOPT_GENE` Vote: $(0,1,0)$
Division gene	Labels: `GENE`, `DIV_GENE` Vote: $(1,0,0)$
Segregation gene	Labels: `GENE`, `SEG_GENE` Filter: $(0,0,0)$

To capture the concepts of evolutionary heritage (i.e. cell lineage), `FROM` and `WAS` relationships between `CELL` systems are used. These indicate heritage of regular cell division and heritage of division in which missegregation occurs respectively. Other metrics essential in tracing the ancestry of evolved cells are recorded using the cell properties: `start`, indicating the iteration in which a cell first came to exist; `nDivs`, recording the number of times a cell has divided; and `missegregationAt`, indicating the iteration in which chrosomome missegregation has occured.

Organisation

Figure 5 shows the hierarchy of a tissue system composed of cells with chromosome distribution B. The `TISSUE` system groups the contained cells to provide an entry point for the recursive algorithm. The horizontal (in the context of the hierarchy tree) `FROM` and `WAS` relationships are created between cell systems as shown in Figure 7.

[2] All filters, votes, and vote functions are set to the defaults unless otherwise stated.

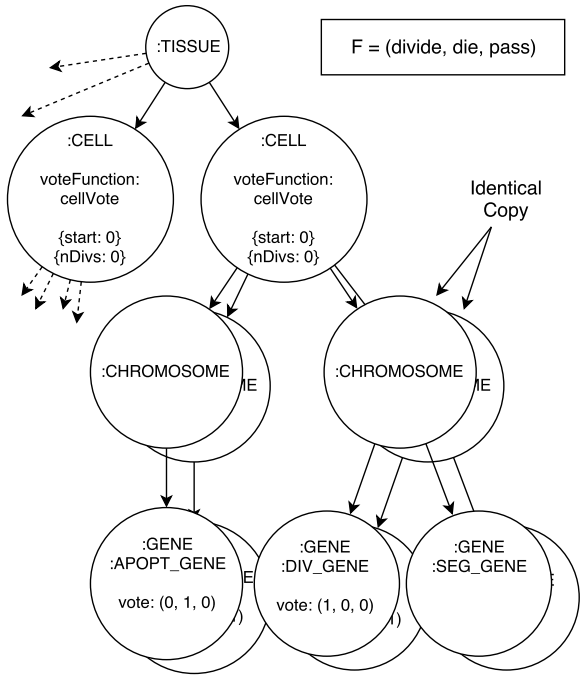

Figure 5: IHDN Tissue to Gene Model. Apoptosis and division gene systems influence function selection according to their votes. The presence of segregation genes is queried during cell division in order to calculate the required probability of missegregation.

Interaction

While there are three cell behaviours to model in this simulation (cell division, cell death, and chromosome missegregation), since missegregation may only occur in the context of a division, it is not treated as a distinct function, rather it is incorporated into the division function. To enable cells to abstain from any behaviour in a given iteration, a `pass` function is also given resulting in the ordered set

$$F = \{\text{divide}, \text{die}, \text{pass}\}$$

The `divide` function performs a `deepClone` of the current system, such that any contained systems are recursively copied, and any incoming `CONTAINS` relationships are also copied (Figure 2). The result is an exact copy of the structural hierarchy and composition, without duplication of any user-defined relationships (i.e. the `FROM` and `WAS` relationships). After cloning, assuming no missegregation has occurred, the `start` property of the clone is set to the current iteration and the `nDivs` property in each system incremented.

For the `die` function, a `inactiveAt` property is set to the current iteration and the system is labelled `INACTIVE` to exclude it from further computation.

The `cellVote` vote function ensures that the probability of selecting the `pass` function is always 0.5, independent of the number of contained genes or their configuration (Figure 6). The remaining 0.5 is then shared (as per the default behaviour) between the `divide` and `die` functions

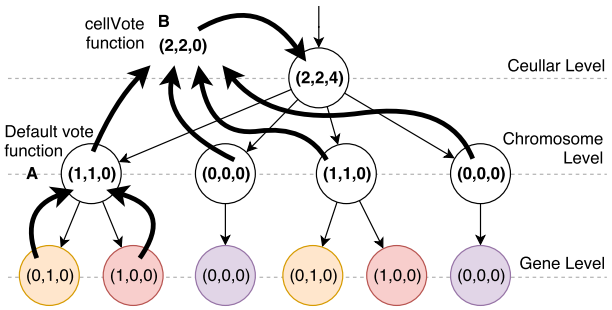

Figure 6: Demonstration of message passing system from lower to higher scale systems in a diploid cell of Distribution A: (A) the votes of the gene systems are summed with the chromosome system's own vote resulting in $(1, 1, 0)$; (B) the `cellVote` function intercepts the votes from the chromosomes, reduces them with the default vote function, but then assigns the total of all elements to the position corresponding to the `pass` function.

proportionally to the number of votes for each, resulting in the following probability distribution for cell function selection:

$$P(f = \text{pass}) = \frac{1}{2}$$
$$P(f = \text{divide} | f \neq \text{pass}) = \frac{d}{a+d}$$
$$P(f = \text{die} | f \neq \text{pass}) = \frac{a}{a+d}$$

where a and d are the number of contained `ADOPT_GENE` and `DIV_GENE` systems respectively.

Since presence of the abstracted segregation gene does not bias function selection, but rather influences the extent to which the chromosomes are correctly segregated during cell division, the number of contained `SEG_GENE` systems is queried via a Cypher call during the `divide` function directly without need for voting or an additional function. As with (Araujo, 2013), the conditional probability of cell missegregation used is[3]:

$$P(\text{missegregation} | f = \text{divide}) = \frac{1}{100} \times (4 - s)$$

where s is the number of contained `SEG_GENE` systems.

In the case that a cell division is subject to chromosome missegregation, a copy of the configuration prior to division is made. The resulting system's `CELL` label is replaced with `CELL_COPY` and `INACTIVE` to prevent its inclusion in any further computation. The resulting pair of aneuploid cells are each linked to it with a `WAS` relationship (see Figure 7). The iteration in which the missegregation occurred is recorded with the `missegregationAt` property on the `CELL_COPY` system.

The `TISSUE` system is the single `ROOT` system, thus every contained system is visited for computation once per iteration.

[3] s may only change ± 1 per division, and only when $P(\text{missegregation}) > 0$, it is therefore guaranteed that $0 \leq s \leq 4$.

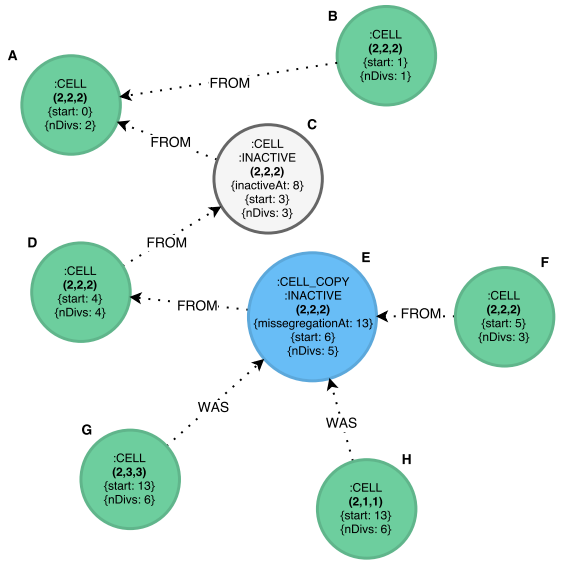

Figure 7: Example simulation output graph demonstrating connections between systems on the cellular level of abstraction, where $(\mathbf{a}, \mathbf{d}, \mathbf{s})$ represents the number of contained apoptosis, division, and segregation genes respectively. This particular graph implies the sequence of events: A divides producing B, A divides producing C, C divides producing D, D divides producing E, C dies, E divides producing F, E divides but missegregates resulting in G and H.

Experiment

To verify our model we test the tumour simulation against the results of (Araujo, 2013) by comparing the tissue size, genome diversity, and ratio of apoptosis to division genes for each of the three chromosome distributions. Then, as a demonstration of the additional capabilities of our model, we analyse the evolutionary heritage (i.e. the cell lineage) of the most prolific aneuploid tumour cells that arise during the simulations.

The simulation is started with 100 identical diploid cells, and executed 20 times for each of the three possible chromosome configurations (Figure 4). The simulation is executed until the tissue exceeds $7,000$ living cells (cell count is monitored at the end of each complete iteration), or 100 iterations are reached (whichever occurs first).

Results

Verification

The simulation results (Figure 8 to 10) of the IHDN model show the expected growth behaviours as demonstrated in the reference model (Araujo, 2013)[4]. When the apoptosis and division genes are distributed in the same chromosome (Distribution A), we see an expected homoeostasis in the size of the tissue. However, when the cells are able to evolve the number of contained copies of the apoptosis and division genes independently, we see the tissue grow in size exponentially. This behaviour is due to the evolution of cells that are 'fitter' (i.e. more prolific and less likely to die) than the initial population of diploid cells. Thus, we see the growth of a tumour.

While Distributions B and C both demonstrate exponential growth, it is observed in the reference model that the

[4]Graphs of reference model simulation are reproduced here with permission and original data from the author.

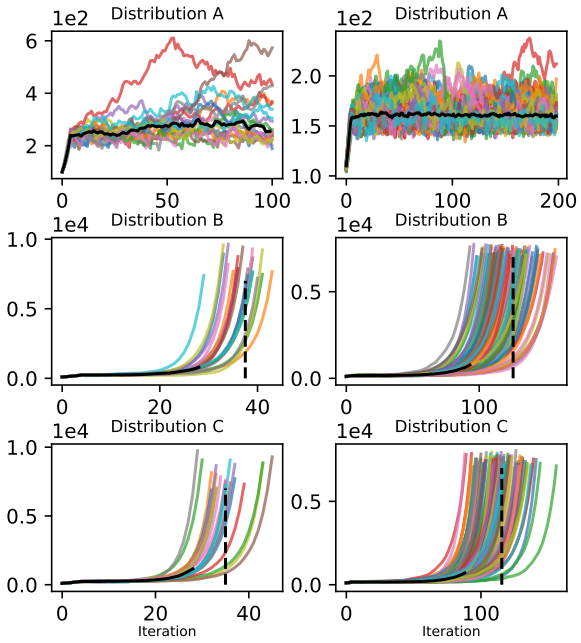

Figure 8: Tissue Size – Total number of living cells per iteration (left: IHDN, right: Reference model). Dashed line indicates median iterations to tissue size $> 7,000$ cells.

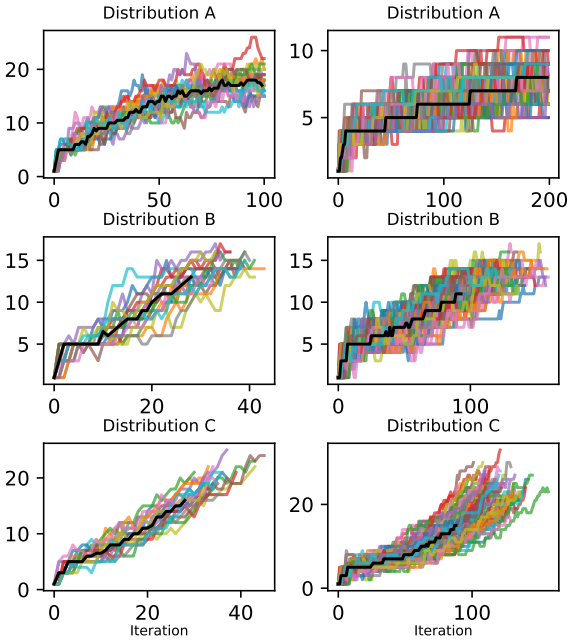

Figure 9: Chromosome Diversity – Number of distinct genome types per iteration (left: IHDN, right: Reference model).

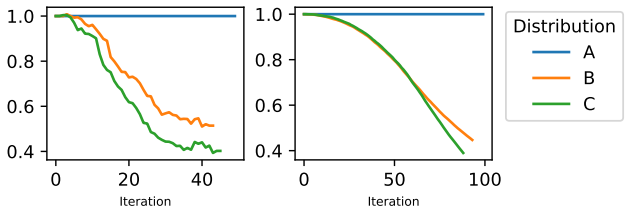

Figure 10: Mean ratio of apoptosis to division genes (left: IHDN, right: Reference model).

rate of growth is faster in C (Araujo, 2013). By comparing the mean number of iterations until the tissue size exceeds $7,000$ cells (dashed vertical line in Figure 8), we observe the same result[5].

Cell Lineage

Having verified the behaviour of our model against an existing conventional implementation, we now demonstrate its advantages. Throughout the remainder of this section, (a, d, s) denotes the number of contained apoptosis, division, and segregation regulatory genes respectively.

Using Neo4j Cypher queries to search the graphs for particular patterns, the complete evolutionary paths of any cell can be traced. Listing 1 demonstrates an example query (visual result in Figure 11) to show three evolutionary paths from the start cell configuration, $(2, 2, 2)$, to the highest occurring cell configuration (a particularly harmful cell) of Distribution C, $(0, 2, 0)$.

Listing 1: Example Cypher query to return three distinct paths of genome evolution from the initial $(2, 2, 2)$ to the cancerous $(0, 2, 0)$.

```
match (c:CELL) where (not (c:INACTIVE)) and
    ↪ c.genome=[0,2,0]
with c match p = (c)-[:FROM|WAS*]->(o)
  where
    ((o:CELL) or (o:CELL_COPY))
    and not (o)-[:FROM|WAS]->()
    and not (c)<-[:FROM|WAS]-()
return p limit 3
```

Going further, for each of the configurations we query all distinct genome evolutionary paths to each of the arising cell configurations, where consecutive matching genomes are removed from the returned sequences. We see that approximately two thirds of the $(0, 2, 0)$ cells of Distribution C followed the simplest possible route (Table 2); because of the higher probability of chromosome missegregation in these cells, many demonstrate increased exploration and oscillation between configurations in their genome ancestry.

[5]Note that due to the difference in computing styles, the iteration numbers do not correspond; however, the purpose here is to validate the life-like evolutionary behaviour of the tissue and cells, not the precise figures of the reference model.

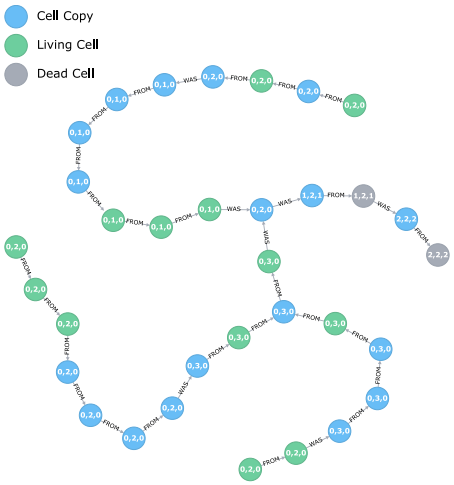

Figure 11: Visual result of the example query (Listing 1). (Graphic produced by the Neo4j Bowser).

However, for the most prolific arising cell configuration of Distribution B, $(0, 2, 2)$, it can be seen that a much greater proportion took the shortest evolutionary path, as the probability for missegregation, and hence evolution, in these cells is much lower.

Distribution B	
Path	%
(2,2,2),(1,2,2),(0,2,2)	86.3
(2,2,2),(1,2,2),(1,1,1),(0,1,1),(0,2,2)	3.81
(2,2,2),(1,2,2),(0,2,2),(0,1,1),(0,2,2)	3.54
Distribution C	
Path	%
(2,2,2),(1,2,1),(0,2,0)	67.3
(2,2,2),(1,2,1),(0,2,0),(0,1,0),(0,2,0)	7.47
(2,2,2),(1,2,1),(0,2,0),(0,3,0),(0,2,0)	7.32

Table 2: Proportion of most abundant final cell configuration that followed the most commonly occurring distinct evolutionary paths.

By considering the mean number of distinct paths, we also see that ancestry of the arising $(0, 2, 0)$ genome configurations (Distribution C) is the most diverse (12.45) across all distributions, followed closely by $(0, 3, 0)$ with 12.05. While these forms of analysis require no additional tooling with IHDNs, they were not possible in previous implementations and could help inform the open debate (for example, (Greaves and Maley, 2012) and (Sottoriva et al., 2015) are two opposing views) over the evolution of such cells.

Conclusion

We have introduced the IHDN model for simulating complex dynamic systems, and verified the effectiveness of its novel, cross-scale message passing system in capturing the

dynamic hierarchical dependencies of living systems.

Having demonstrated its application in simulating aneuploid tumour development we observe the expected growth behaviours for all three chromosome distributions. We have also shown that through integration with a graph database the IHDN model facilitates powerful 'out of the box' analysis not possible in prior models, demonstrated here through tracing the evolutionary paths of arising cell configurations.

Moreover, the pattern matching techniques we have demonstrated are not restricted to post-analysis; any system functions may take full advantage of the optimised pattern matching Neo4j Cypher query engine during execution, thus enabling systems to interact or adapt their behaviour according to the detection of complex network structures.

Acknowledgements

Arturo Araujo – for his advice on the biological model presented, as well as providing his original simulation data.
Braintree Ltd – for providing the funding to support this work.

References

Ahmed, S., Karsiti, M. N., and Agustiawan, H. (2006). A Development Framework for Collaborative Robots Using Feedback Control.

Araujo, A. (2013). *Modelling Chromosome Missegregation in Tumour Evolution*. PhD thesis, UCL.

Arel, I., Liu, C., Urbanik, T., and Kohls, A. (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems*, 4(2):128.

Axelrod, R. (1997). *The complexity of cooperation: Agent-ased models of competition and collaoration*.

Batty, M., Xie, Y., and Sun, Z. (1999). Modeling urban dynamics through GIS-based cellular automata. *Computers, Environment and Urban Systems*, 23(3):205–233.

Bentley, P. J. (2009). Methods for improving simulations of biological systems: systemic computation and fractal proteins. *Journal of The Royal Society Interface*, 6(Suppl_4):S451–S466.

Bolton, H., Graham, S. J., Van Der Aa, N., Kumar, P., Theunis, K., Gallardo, E. F., Voet, T., and Zernicka-Goetz, M. (2016). Mouse model of chromosome mosaicism reveals lineage-specific depletion of aneuploid cells and normal developmental potential. *Obstetrical and Gynecological Survey*, 71(11):665–666.

Bortot, L., Auchmann, B., Garcia, I. C., Fernando Navarro, A. M., Maciejewski, M., Mentink, M., Prioli, M., Ravaioli, E., Schops, S., and Verweij, A. (2017). STEAM: A Hierarchical Co-Simulation Framework for Superconducting Accelerator Magnet Circuits. *IEEE Transactions on Applied Superconductivity*, 28(3).

Cremer, T., Cremer, C., Not at Dartmouth/Dhmclibraries, and on interlibrary loan, R. (2001). Chromosome territories, nuclear architecture and gene regulation in mammalian cells. *Nature Reviews Genetics*, 2(4):292–301.

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Complexity*, 4(5):41–60.

Frank, S. a. (2007). Cell Lineage History. In *Dynamics of Cancer. Incidence, Inheritance, and Evolution*, chapter 14, pages 1–378. Princeton University Press.

Funk, S., Salathé, M., and Jansen, V. a. a. (2010). Modelling the influence of human behaviour on the spread of infectious diseases: a review. *Journal of the Royal Society, Interface / the Royal Society*, 7(50):1247–56.

Gilbert, N. (2004). Agent-based social simulation: dealing with complexity. *The Complex Systems Network of Excellence*, 9:1–14.

Gonzalez, M. W. and Kann, M. G. (2012). Chapter 4: Protein Interactions and Disease. *PLoS Computational Biology*, 8(12).

Greaves, M. and Maley, C. C. (2012). Clonal evolution in cancer.

Gross, T., D'Lima, C., and Blasius, B. (2006). Epidemic Dynamics on an Adaptive Network. *Physical Review Letters*, 96(20):208701.

Lenaerts, T., Groß, D., and Watson, R. (2002). On the Modelling of Dynamical Hierarchies : Introduction to the Workshop. *Proceedings of the Alife VIII workshop*, (31):37.

Pearlmutter, B. A. and Houghton, C. J. (2009). A New Hypothesis for Sleep: Tuning for Criticality. *Neural Computation*, 21(6):1622–1641.

Sakellariou, C. and Bentley, P. J. (2015). Demonstrating the performance , flexibility and programmability of the hardware architecture of systemic computation modelling cancer growth. *Bio-Inspired Computation*, 7(6).

Sayama, H. (2007). Generative network automata: A generalized framework for modeling complex dynamical systems with autonomously varying topologies. *Proceedings of the 2007 IEEE Symposium on Artificial Life, CI-ALife 2007*, pages 214–221.

Sayama, H., Pestov, I., Schmidt, J., Bush, B. J., Wong, C., Yamanoi, J., and Gross, T. (2013). Modeling complex systems with adaptive networks. *Computers and Mathematics with Applications*, 65(10):1645–1664.

Sottoriva, A., Kang, H., Ma, Z., Graham, T. A., Salomon, M. P., Zhao, J., Marjoram, P., Siegmund, K., Press, M. F., Shibata, D., and Curtis, C. (2015). A big bang model of human colorectal tumor growth. *Nature Genetics*, 47(3):209–216.

Webster, M. and Malcolm, G. (2008). Hierarchical components and entity-based modelling in artificial life. *Artificial Life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems, ALIFE 2008*, (1991):678–685.

Yao, Y. and Van de Peer, Y. (2017). Simulating Biological Complexity through Artificial Evolution. In *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, pages 1–8. IEEE.