B.Tech Project (Stage I) Report:

Estimating the Dynamics of Large Boolean Networks

Shabnam Sahay
190050111

Supervisor: Prof. Ganesh Viswanathan
Co-supervisor: Prof. Ashutosh Gupta

IIT Bombay

November 2022

1 Introduction

Several biological processes are controlled by networks of interacting genes, proteins, transcription factors, signalling molecules, and other biological entities. Modelling and performing simulations on these networks can aid in obtaining an understanding of the conditions that drive their dynamics to certain states - for example, the differences in conditions that drive a p-53 activated cell to cell-cycle arrest versus apoptosis.

Boolean dynamic networks are a commonly used model for such biological networks. When modelling them in boolean form, each entity is represented by a single node which can either have an ON (1) or OFF (0) value. The value of each entity at a given time is determined by the value of its 'neighbouring' entities which interact with it, either activating or inhibiting it. Such relationships can be encoded into a boolean logic function that takes as input the values of these other entities and outputs the value of the entity in consideration.

Extensive work has been done to construct such boolean models reflecting biological phenomena and perform simulations to determine attractors, steady state probabilities, and other metrics which can help uncover the behaviour and dynamics of the real biological systems. However, the accuracy of the conclusions drawn from the simulations depends heavily on how closely the definition of the model matches the true biological network.

Incorporating a larger number of nodes and more complex interactions between them (reflected by a larger in-degree of nodes in the network) will bring the model closer to the true biological network, boosting the accuracy of the inferences made from simulations done with the model. The problem that arises here is that since the state transition graph of the network grows exponentially with the number of nodes contained, large networks demand heavy computational power and are computationally intractable beyond a point.

Thus, a need arises to design approaches to estimate the partial state transition graph of such large networks in a way that allows recapitulation of the dynamics of the entire state transition graph, without having to generate it (which would take exponential time). In this report, a brief introduction to Boolean dynamic modelling will be given, following which the implementation of a few such approaches with the observed outcomes will be described.

2 Boolean dynamic networks

2.1 Notation

Consider a network B with n nodes $a_1, a_2, \ldots, a_n$. Each node a_i has a corresponding boolean value which varies as a function of time as $v_i(t)$. $\forall t \in \mathbb{N}$ and $\forall i \in \{1, 2, \ldots, n\}$, $v_i(t)$ can take either the value 0 or 1. The state of the network B at time t is uniquely represented by the binary string $V(t) = v_1(t)v_2(t)\cdots v_n(t)$.

$v_i(t)$ is determined by a boolean function f_i that takes as input the values of a set M_i of m_i nodes at time t'. While the time t' is dependent on the update scheme used (explained in more detail below), f_i itself has a specified functional form that is independent of the update scheme.

$$v_i = f_i(M_i) = f_i(v_k \forall a_k \in M_i) \text{ where } M_i = \{a_{i1}, a_{i2}, \ldots, a_{im_i}\} \tag{1}$$

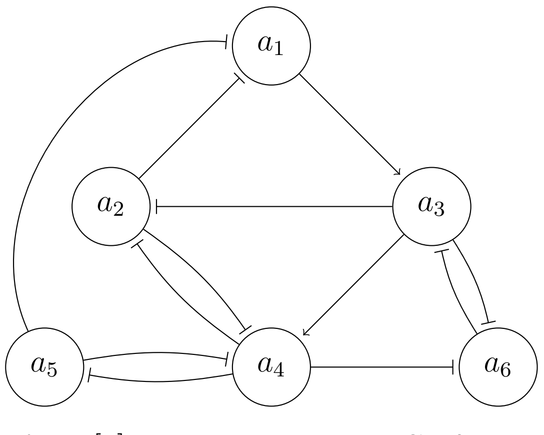

Figure 1: A six-node network from [1], corresponding to TGFβ signalling in cancer cells. Nodes $a_1, a_2, a_3, a_4, a_5, a_6$ denote TGFβ, miRNA200, Snail1, Zeb1, Ovol2, and miRNA34a respectively. A $\rightarrow$ connection represents activation and $\dashv$ connection represents inhibition.

f_i	M_i	$f_i(M_i)$
f_1	$\{a_2, a_5\}$	$\neg v_2 \vee \neg v_5$
f_2	$\{a_3, a_4\}$	$\neg v_3 \vee \neg v_4$
f_3	$\{a_1, a_6\}$	$v_1 \wedge \neg v_6$
f_4	$\{a_2, a_3, a_5\}$	$v_3 \wedge \neg v_2 \wedge \neg v_5$
f_5	$\{a_4\}$	$\neg v_4$
f_6	$\{a_3, a_4\}$	$\neg v_3 \vee \neg v_4$

Table 1: f_i, M_i and $f_i(M_i)$ for the 6-node network shown in 1.

2.2 Update schemes

$V(t)$ depends on the value of each of its constituent nodes. Depending on whether these values are updated simultaneously (all nodes together) or step-by-step (one node after another), there are three update schemes commonly used in boolean dynamic networks:

1. Synchronous update

   All $v_i(t)$ values are computed in a simultaneous update, by considering the values of all the nodes in their respective M_i sets at time $t-1$.

$$v_i(t) = f_i(v_{i1}(t-1), v_{i2}(t-1), \ldots, v_{im_i}(t-1)) \forall i \in \{1, \ldots, n\} \tag{2}$$

2. Ordered asynchronous update

   Let the period of time between $t-1$ and t be broken down into n smaller equally-spaced intervals, separated by the time markers $t_1, t_2, \ldots, t_n$ where $t_n = t$. Let $t_0 = t-1$.

   v_1 is updated at time t_1 using the values of all nodes as they originally are at $t_0 = t-1$. Thus at t_1, v_1 has an updated value, while the values of all other nodes simply carry over to remain the same as they were at t_0.

   Next, v_2 is updated at time t_2 using the values of all nodes as they are at t_1. Thus, if $a_1 \in M_2$, v_2 is updated using the previously updated value v_1.

Similarly, at each time t_i, v_i is updated using the values of all nodes as they are at t_{i-1}, and uses the already updated values of nodes $a_1, \ldots, a_{i-1}$ to do so.

Thus the value of the nodes is updated step-by-step here in a well-defined order.

3. Random-order asynchronous (ROA) update

   The difference between this update scheme and the ordered asynchronous update scheme is only in the order in which the nodes are updated step-by-step. Whereas the latter has a well-defined update order, here a random ordering of the nodes is chosen in which to perform the updates.

   With n nodes in the network, there are hence $n!$ different possbile update orders which can be chosen for every update from $V(t-1)$ to $V(t)$.

2.3 Measures of network dynamics

A state transition graph (STG) captures the one-step transition probabilities between all possible pairs of states of the network. A network with n nodes will have 2^n possible states, hence an adjacency matrix representation of an STG will be a $2^n \times 2^n$ matrix S. S_{ij} then denotes the probability of a one-step transition from the ith state to the jth state occurring.

Clearly, these probabilities will be dependent on the update scheme used. A synchronous update scheme defines a single possible end-state for a one-step transition from a given start-state of the network, leading to the construction of a binary-valued S.

However, an ROA update scheme, which considers $n!$ possible different orders in which to update the values of the nodes, usually leads to more than one possible end-state for a one-step transition from a given start-state. In this case, S_{ij} is calculated by finding the number of permutations (update orders) that lead to state j from state i, and dividing this by $n!$.

Every Boolean network has particular states called fixed points or attractors dependent on the update scheme being utilised. Updating the network when it is in one of these states will result in it remaining in the same state, i.e. all possible one-step transitions from that state lead to itself. Hence the network is said to attain steady state when it reaches any attractor.

The steady state probability (SSP) of an attractor is the probability that, if the network is initialised to a random state, it will attain steady state in that particular attractor. Determining the SSPs of various attractors of a network, as well as its STG, can be useful in understanding and making further inferences about its behaviour.

The aim is hence to generate partial STGs of a large network using limited states (and permutations, in the case of an ROA update scheme), which can recapitulate the dynamics encoded by the complete STG, and hence enable the analysis of the network in an efficient manner.

3 Limiting permutations for an ROA update scheme

3.1 Constructing STGs with forward-transition ROA updates

An ROA update scheme is likely to be more biologically relevant than a synchronous one, since all participating entities in the network are unlikely to update their value at exactly the same instant in time. Moreover, the order in which they will update in a real system is not deterministic, making ROA update more relevant than an ordered asynchronous update.

Given a network B with n nodes, the state space of the network is 2^n. To construct a complete $2^n \times 2^n$ STG for this network with an ROA update scheme, the total number of permutations to be considered for each one-step transition is $n!$. Since $n!$ grows much faster than 2^n (Figure 2), the bottleneck of constructing the STG with ROA updates at large n becomes the consideration of all possible permutations for each one-step transition.

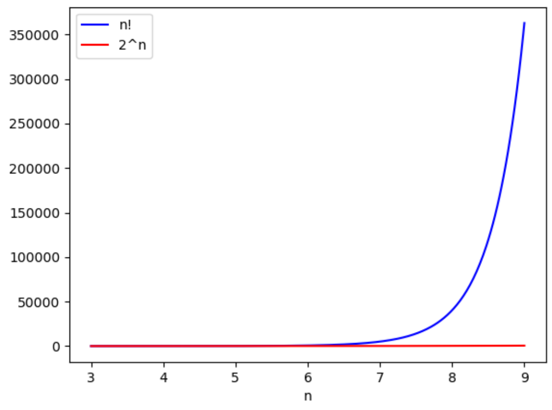

Figure 2: The number of permutations required to consider all possible orders in an ROA update scheme ($n!$) grows much faster than the state space of a network with n nodes (2^n).

A possible approach to tackle this is to consider only a subset of these $n!$ permutations, working under the assumption that increasing the number of permutations used beyond a point will not significantly change the calculated transition probabilities of the network.

Algorithm 3.1: Algorithm to construct the STG of a network B using ROA update scheme with a number of permutations q.

```
Data: $B$, $q$
Result: $S^q$, an estimated STG of $B$
$q \leftarrow 0$
List of permutations utilised so far $L \leftarrow$ empty list
$S^q \leftarrow$ empty STG (an $n \times n$ matrix of zeroes) ;
foreach $i$ in $1 : q$ do
    Add a new permutation $p$ of nodes in $B$ to $L$
    Update $S^q$ with one-step transitions obtained by asynchronous update in order of $p$
end
return $S^q$
```

To verify this approach, it was applied to a small, computationally tractable network - the one demonstrated in Figure 1. The generated approximate STG by applying Algorithm 3.1 to this network with $q = 360$ was compared to the complete STG obtained by applying the algorithm to the network with $q = 6! = 720$.

These two matrices are visualized as heatmaps in Figures 3 and 4, where they can be seen to have high similarity. This observation is supported by the Frobenius norm FB_{norm} for the STG constructed with $q = 720$ being 5.006889, while that constructed with $q = 360$ has $FB_{norm} = 5.012367$. The difference of the two STGs has $FB_{norm} = 0.163334$.

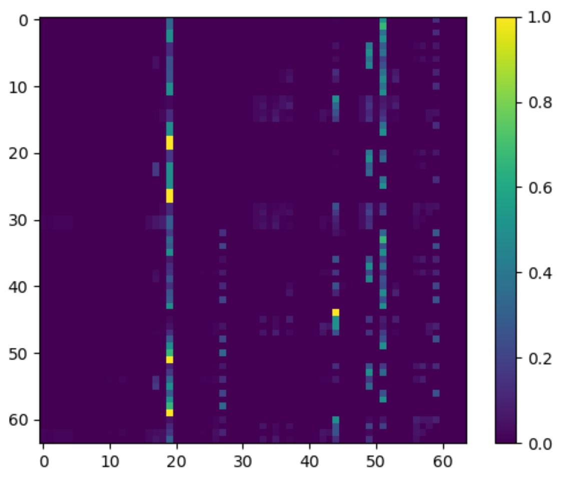

Figure 3: The true STG (as a heatmap) of the network shown in Figure 1, generated by running 3.1 with $q = 720$. $FB_{norm} = 5.006889$.

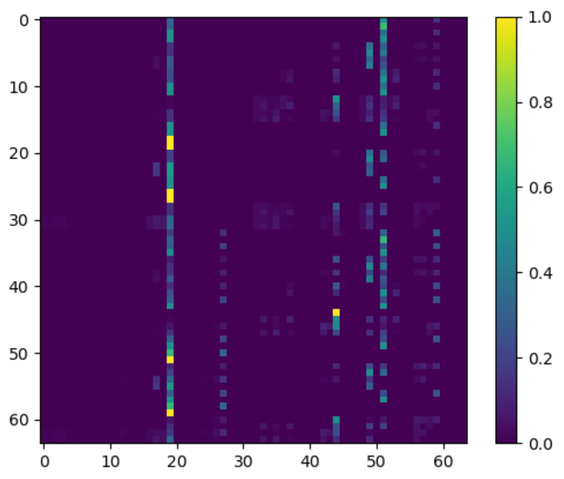

Figure 4: The estimated STG (as a heatmap) of the network shown in Figure 1, generated by running 3.1 with $q = 360$. $FB_{norm} = 5.012367$.

This approach thus proves to be useful in constructing a partial STG that approximates the complete STG to some degree of accuracy. This will be revisited in a subsequent section of this report in an extended form - that of the BM-ProSPR algorithm.

3.2 Constructing STGs with reverse-transition ROA updates

Given a particular state of the network, one can find all possible previous states that would have generated this state following a particular update scheme. If the ROA scheme is used, then the calculation of the back-transition corresponding to each of the $n!$ possible permutations can be done for a given state, in order to find the set of its precursor states.

Continuing with the principle of using only a subset of all possible permutations in order to maintain computational tractability for large networks, Algorithm 3.2 was applied to the same 6-node network 1 to construct its STG using reverse-transitions. The two known fixed points of the network, 010011 and 101100, were the only initial states provided as the input L.

Algorithm 3.2 was run for this network with $q = 720$ to enable reverse-construction of the complete STG - the generated STG is identical to the one shown in Figure 3. Another run was performed with $q = 360$, to reverse-construct an estimate of the STG.

These two STGs are visualized as heatmaps in Figures 5 and 6 respectively, where they can be observed to have high similarity, supported by their respective Frobenius norms.

Absorption probability matrices were also calculated from each of these STGs separately following the procedure detailed in [2]. These matrices have the dimensions (no. of fixed points $\times$ no. of transient states) of the network. When the difference of these two absorption probability matrices was taken, the Frobenius norm of the resulting matrix obtained was 0.081590.

However, using a reverse-transition approach with the ROA update scheme to construct a partial STG has scalability issues. For a network with n nodes, the running time for Algorithm 3.2 increases much faster with n than that for Algorithm 3.1 does.

Algorithm 3.2: Algorithm to construct the STG of a network B using reversed ROA update scheme with a number of permutations q.

```
Data: $B$, $q$, $L$ (a list of known network states for starting calculation of back-transitions)
Result: $S^q$, an estimated STG of $B$
$S^q \leftarrow$ empty STG (an $n \times n$ matrix of zeroes)
$K \leftarrow$ empty list (tracks the states whose back-transitions have been calculated so far)
while $L$ is not empty do
    $V \leftarrow$ the first element of $L$
    $P \leftarrow q$ randomly generated permutations of the nodes in $B$
    foreach $p$ in $P$ do
        Identify a set of states $T$ which, when updated according to $B$'s rules in the order of
          permutation $p$, give rise to state $V$
        foreach $t$ in $T$ do
            Update $S^q$ corresponding to a $t \rightarrow V$ transition
            if $t$ not in $L$ and $t$ not in $K$ then
                Append $t$ to the end of $L$
            end
        end
    end
    Delete $V$ from $L$ and add $V$ to $K$
end
return $S^q$
```

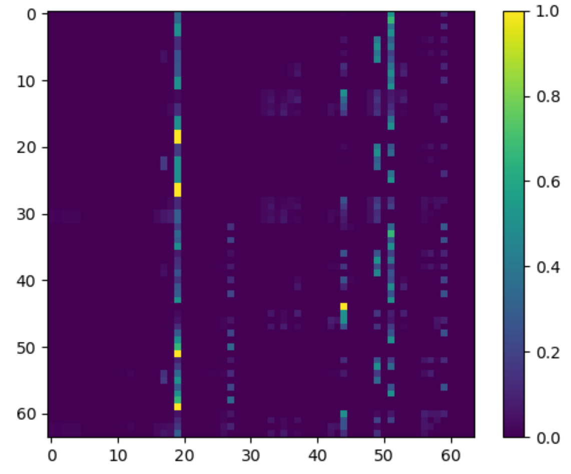
Figure 5: The true STG (as a heatmap) of the network shown in Figure 1, generated by running 3.2 with $q = 720$. $FB_{norm} = 5.006889$.

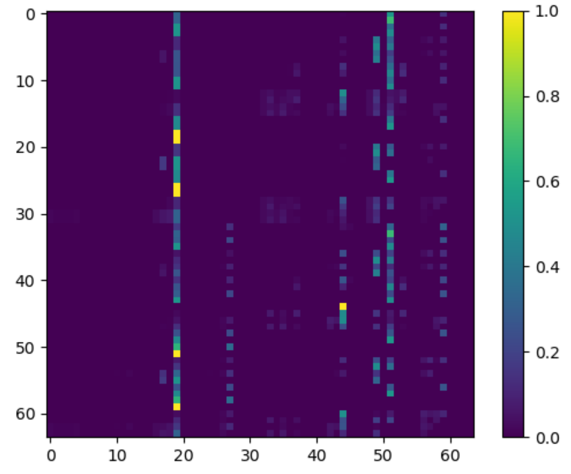
Figure 6: The estimated STG (as a heatmap) of the network shown in Figure 1, generated by running 3.2 with $q = 360$. $FB_{norm} = 4.994795$.

The principle itself of utilising reverse transitions also poses an issue. To construct a partial STG with this method, one must supply the attractors of the network as L to Algorithm 3.2. Attractors will have a very large in-degree, since multiple transient states will connect to a single attractor via one-step transitions. Hence, limited permutations are unlikely to capture the large amount of out-branching encountered when reverse transitions from the attractors are being computed, and thus are unlikely to recapitulate the dynamics of the network accurately.

3.3 The BM-ProSPR algorithm

Constructing the partial STG with forward-transitions computed for limited permutations appears to have more utility than doing so with reverse-transitions. The next problem that then arises is the estimation of the number of permutations q to be supplied as input to Algorithm 3.1.

The BM-ProSPR algorithm [3] provides a solution to this. It computes a minimum number of permutations Q, with which when we perform ROA on the given network from all of its starting states, generates an STG that is a reasonable estimate of the true STG of the network.

Briefly outlining the algorithm, the steps followed are as shown in Algorithm 3.3. The Q that is returned can be used to construct a partial STG that estimates the dynamics of the network B within the accuracy allowed by parameters ϵ_1 and ϵ_2.

Algorithm 3.3: BM-ProSPR algorithm from [3]. The details of computation of the metrics $\mathbb{T}^q, \mathbb{PR}^q, \mathbb{D}^q$ can be found in the same article.

```
Data: $B$, $\epsilon_1, \epsilon_2, N$
Result: A minimum number of permutations $Q$ required to estimate the STG of the
        network within a given accuracy with a ROA update scheme
foreach $i$ in $1 : N$ do
    $q \leftarrow 0$
    List of permutations utilised so far $L \leftarrow$ empty list
    $S^q \leftarrow$ empty STG; while True do
        Add a new permutation $p$ of nodes in $B$ to $L$
        Update $S^q$ with one-step transitions obtained by asynchronous update in order of $p$
        Calculate metrics $\mathbb{T}^q, \mathbb{PR}^q, \mathbb{D}^q$
        if $\mathbb{T}^q - \mathbb{T}^{q-1} \leq \epsilon_1$ and $\mathbb{D}^q \leq \epsilon_2$ then
            break
        end
    end
    Save $q$ as $q_i$
end
$Q \leftarrow$ average of $q_i \forall i \in \{1, \ldots, N\}$
return $Q$
```

This algorithm was first implemented and applied to the six-node network shown in Figure 7, in order to compare the true STG with the partial STG constructed using the Q estimated by BM-ProSPR.

These STGs are shown in Figures 8 and 9, with the value of Q returned by the algorithm being 193. They can be seen to be highly similar, with Frobenius norms of 6.385733 and 6.381962 for the true STG and estimated STG respectively. When the difference of these two STGs is taken, the resulting matrix has $FB_{norm} = 0.065626$.

The variation of the T^q and D^q metrics computed by the algorithm versus q is demonstrated in Figure 10 for values of q upto 50. A rapid convergence of both metrics to zero can be seen.

Next, we applied the algorithm to a 26-node network described in [4]. Gupta et. al constructed this 26-node dynamic Boolean network in [4] to decipher the regulatory effects of lncRNA GAS5 in gastric cancer. They utilised a synchronous update scheme to simulate the network and make inferences about its dynamics and steady states.

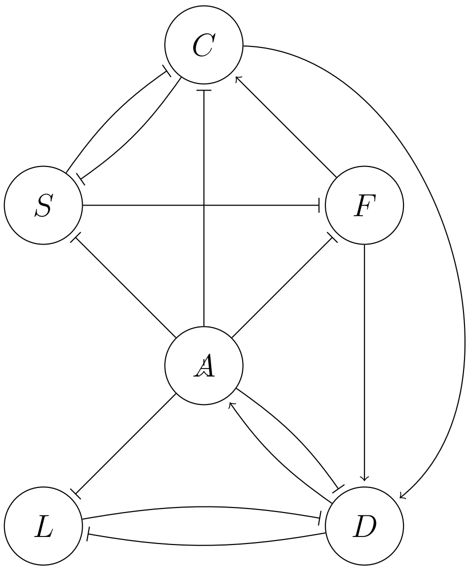

Figure 7: A six-node network from [3], corresponding to T-cell Large Granular Lymphocyte apoptosis. Nodes A, C, D, F, L, and S, denote Apoptosis, Ceramide, DISC, Fas, FLIP, and S1P respectively. A $\rightarrow$ connection represents activation and $\dashv$ connection represents inhibition.

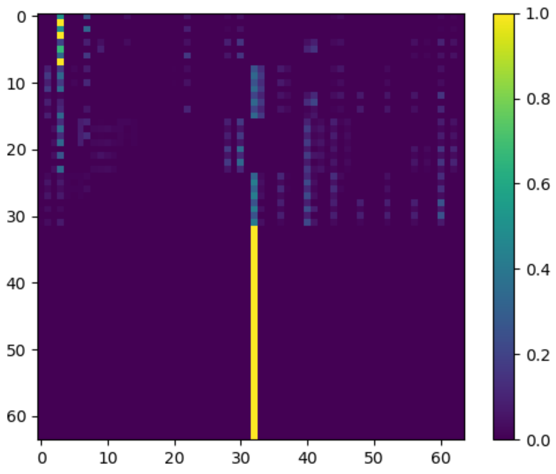

Figure 8: The true STG (as a heatmap) of the network in Figure 7, generated by running ROA updates for each one-step transition over all 720 permutations. $FB_{norm} = 6.385733$.

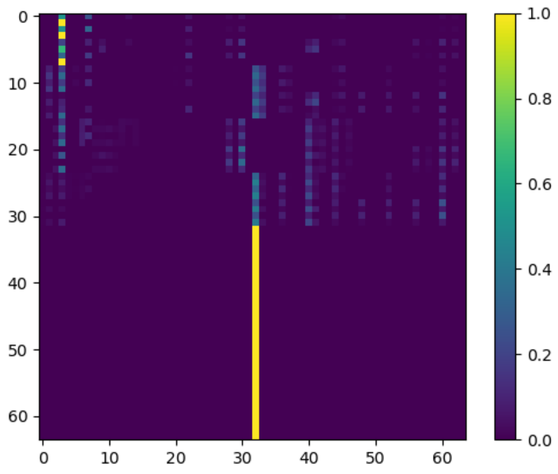

Figure 9: The estimated STG (as a heatmap) of the network in Figure 7, generated by running ROA updates over the BM-ProSPR-given value of 193 permutations. $FB_{norm} = 6.381962$.

Here, we attempt to apply the BM-ProSPR algorithm detailed in [3] to this network to estimate its STG and make inferences from its behaviour under an ROA update scheme. The network has an input node DNA_Damage, which was set to have a constitutive value of 1 (i.e. always ON) during simulations. Figure 11 demonstrates the obtained variation of T^q up to $q = 50$.

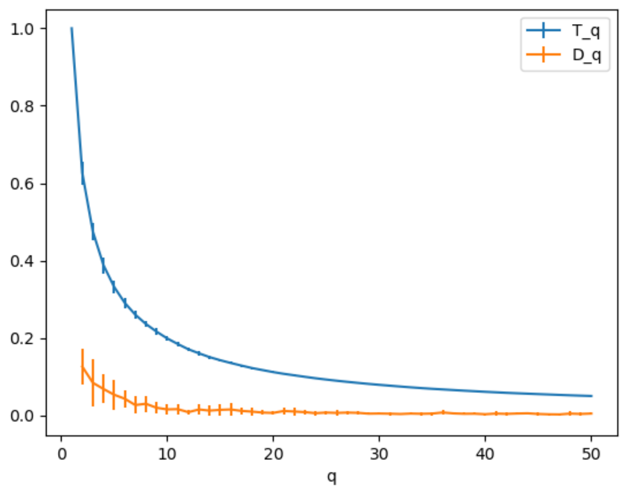

Figure 10: Mean and standard deviation (taken over $N = 20$ iterations) of T^q and D^q versus q plotted up to $q = 50$, for a run of Algorithm 3.3 on the network shown in Figure 7. The value of Q produced by this run was 193, i.e. running ROA updates with 193 permutations out of a total of $6! = 720$ permutations is enough to generate an accurate estimate of the network's STG.

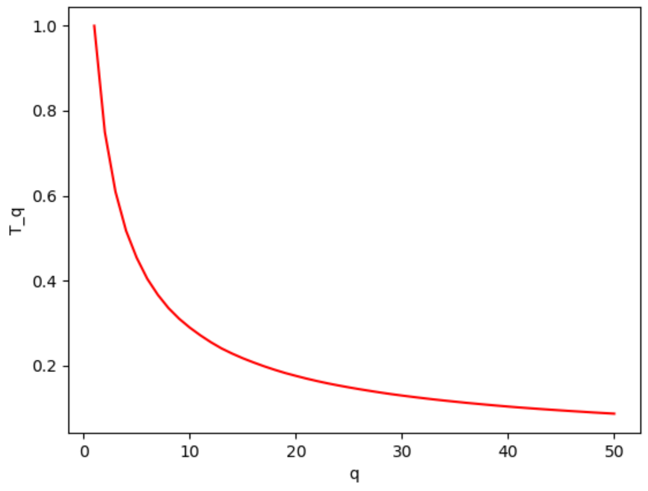

Figure 11: T^q versus q upto $q = 50$ for a run of 3.3 on the network in [4] with $N = 1$.

Calculations of D^q and then Q itself for this network are to be done as a next step, by tackling the high runtimes of PageRank calculations for the $2^{26} \times 2^{26}$ STG required to compute $\mathbb{PR}^q$ and $\mathbb{D}^q$, or using a system with more computational power.

4 Limiting the state space of the network

4.1 Synchronous updates with a subset of states

BM-ProSPR provides a solution to the bottleneck of a large number of permutations being involved in ROA update schemes, by estimating a minimum number of permutations which are sufficient to estimate the dynamics of the network when used to construct a partial STG.

The next problem is then limiting the state space of the network. Though 2^n grows more slowly than $n!$, it still presents a computational challenge for larger values of n. It will hence be useful to be able to identify a subset of all 2^n states with which, when simulations are performed, the derived estimations of the STG and steady-state probabilities are close to their true values.

The synchronous update scheme was utilised for this purpose. Applying the synchronous update scheme on an increasing number of randomly selected initial states of the network, an estimate of the SSPs of the network attractors was obtained as described in Algorithm 4.1.

As the SSPs converge over the increasing number of initial states, a point is reached where their fluctuations remain within a certain limit. A cutoff can be defined here, selecting the minimum number of states required to attain this saturation, and thus the minimum number of states sufficient to generate behaviour representative of the entire state space of the network.

Algorithm 4.1: Algorithm used to estimate steady state probabilities of attractors in a network B using synchronous update scheme.

```
Data: $B$, $\epsilon$
Result: $SSP$, the estimated steady state probability for each detected attractor in $B$
$SSP \leftarrow$ an empty key value list
$initial\_states \leftarrow$ empty list
while True do
    Generate a random initial state $V$ of $B$ not already present in $initial\_states$
    Starting from $V$, perform synchronous updates till a fixed point $F$ is reached
    Add $V$ to $initial\_states$
    Save $SSP$ from the previous iteration as $SSP_{prev}$
    if $F$ already exists as a key in $SSP$ then
        Add 1 to the value of $F$ in SSP, else Add $F$ as a key to $SSP$ and set its value to 1
    end
    if $|SSP\{key\} - SSP_{prev}\{key\}| < \epsilon$ for all keys in $SSP$ then
        break
    end
end
return $SSP$
```

A 40-node network defined in [5] was considered. This network has an input node Transfected_miRNA, which was set to have a constitutive value of 1 (i.e. always ON) in our simulations. Following this, runs of Algorithm 4.1 on this network produced four possible attractors, listed in Table 2 with their corresponding phenotypic meaning.

Fixed point of network	Cellular phenotype
1111100011101100010000110011000000000001	Senescence
1111100011101010000000110111010111000010	Apoptosis
1111100011101010000000000110010111110100	Autophagy
1111100011101100010000000010000000110101	Unknown

Table 2: The fixed points of the 40-node network described in [5], with their corresponding phenotypes. Steady state probabilities are calculated for these fixed points using 4.1.

For all four attractors, there is a clear convergence of their respective SSP values as the number of initial states considered increases, as demonstrated in Figures 12, 13 and 14.

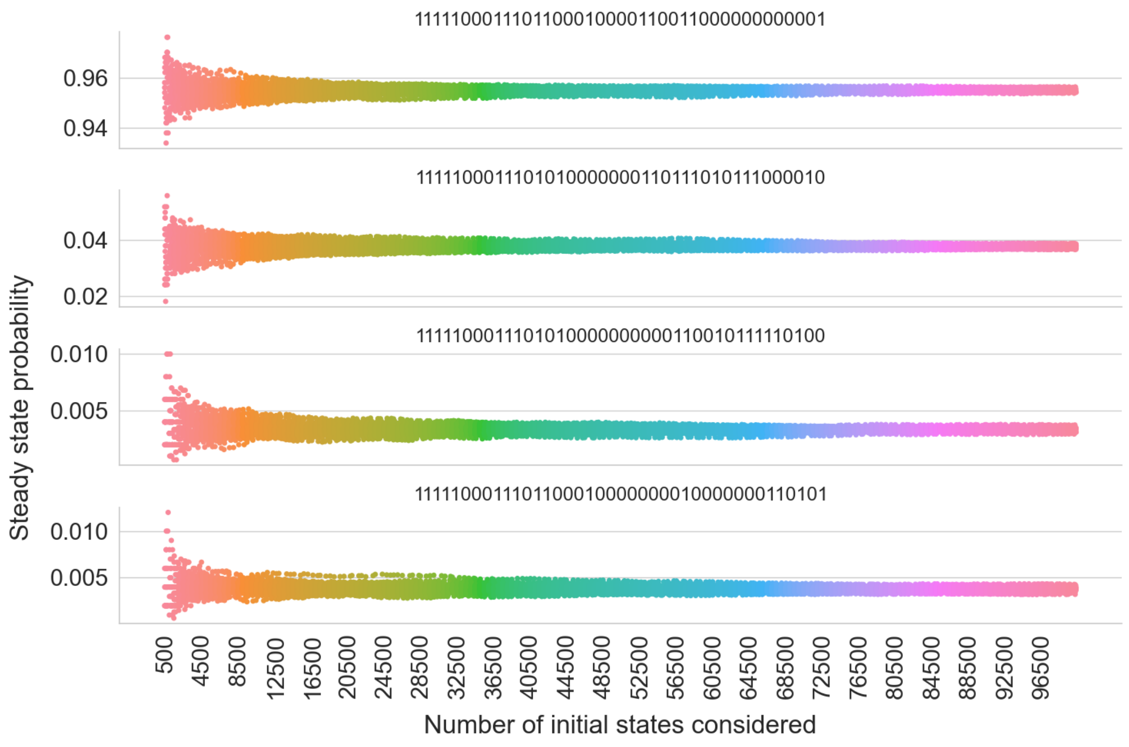

Figure 12: The steady state probabilities (SSPs) of the attractors versus number of initial states considered, obtained by performing 50 runs of Algorithm 4.1 for the 40-node network in [5] upto 10000 initial states.

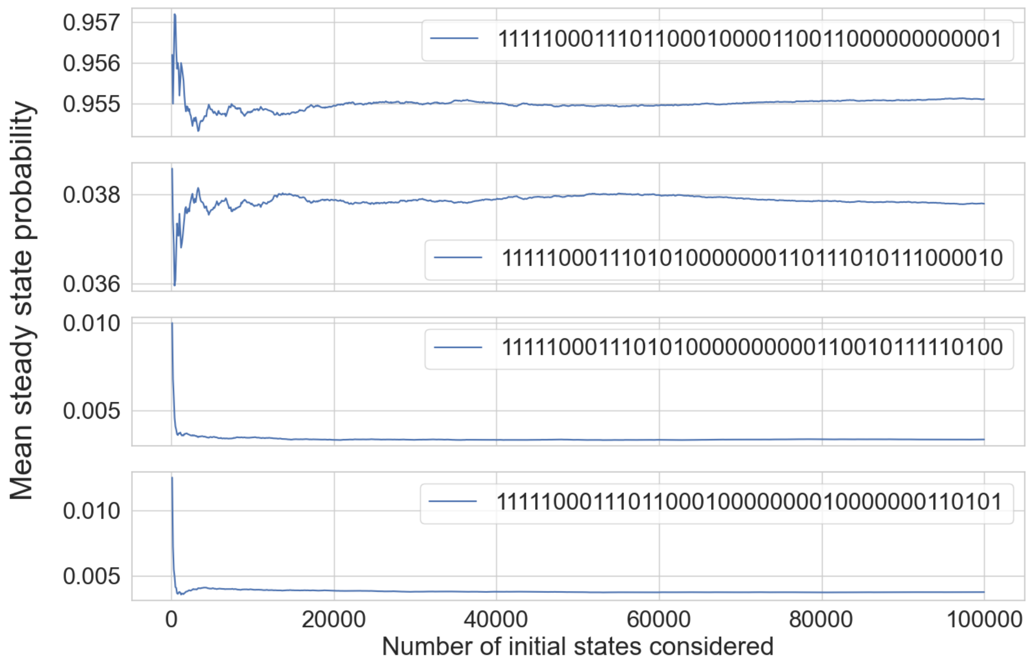

Figure 13: The mean steady state probabilities (SSPs) of the attractors versus number of initial states considered, obtained by averaging across 50 runs of Algorithm 4.1 for the 40-node network in [5] upto 100000 initial states.

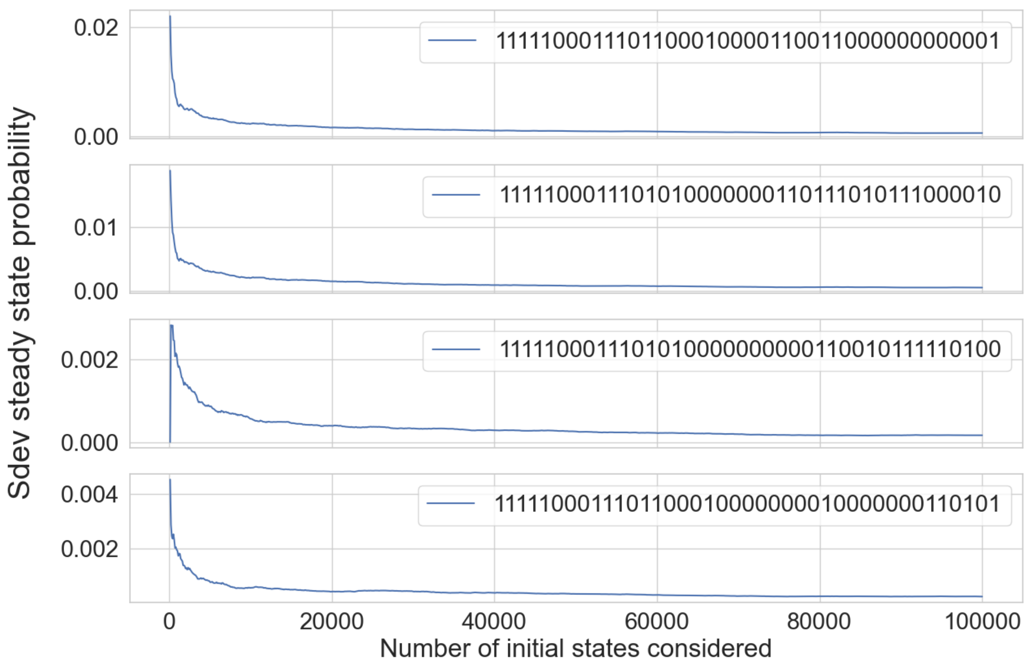

Figure 14: The standard deviation of the steady state probabilities (SSPs) of the attractors versus number of initial states considered, obtained by performing 50 runs of Algorithm 4.1 for the 40-node network in [5] upto 100000 initial states.

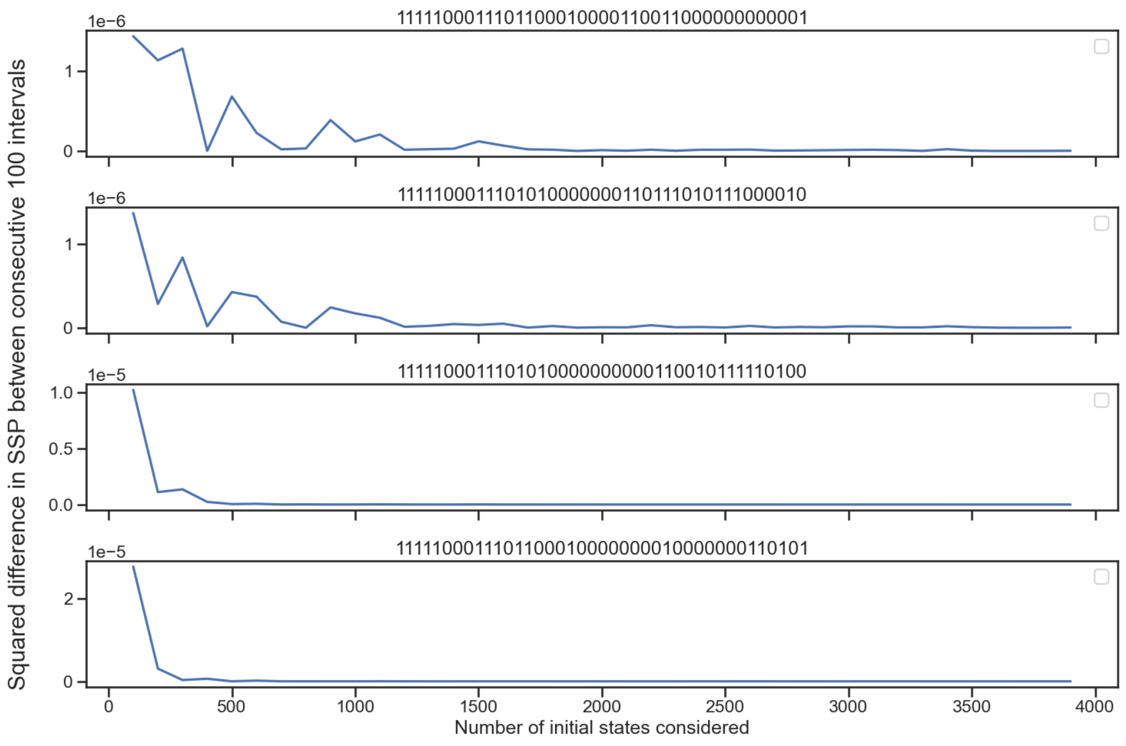

Figure 15: Squared difference between SSP values estimated at consecutive intervals of 100 additional initial states considered, up to a total of 4000 initial states.

The squared difference between SSP values estimated at consecutive intervals of 100 initial states considered is plotted in Figure 15. This error measure converges towards zero very rapidly, flattening out when approximately only 3000 initial states are considered, for a network that has a state space of 2^{40}.

However, convergence is observed at a higher number of initial states considered in Figures 12, 13 and 14. Clearly defining a way to select a cutoff on the number of initial states and thus limit the state space in a way that still allows accurate estimation of the network dynamics then becomes an important next step.

Thus, the approach of using a synchronous update scheme to estimate the steady state probabilities of the attractors of a large boolean network, and consequently derive a limit on the state space of the network, is useful and will be utilised in future work.

4.2 Individual-based mean-field approximation

A second approach to estimate a limit on a network's state space was also implemented.

Consider the probability that node a_i of the network B has a value $v_i = 1$ at a given time t. Let $s_i(t)$ represent this probability. A mean-field approximation to estimate this probability was suggested by Parmer et. al. in [6]. The following analytical expression for $s_i(t)$ is described:

$$s_i(t) = \sum_{j=1}^{2^{m_i}} \delta_{1,f_i(M_i^j)} \prod_{a_k \in M_i} [s_k(t-1)]^{v_k^j} [1 - s_k(t-1)]^{1-v_k^j} \tag{3}$$

This is essentially a sum over all 2^{m_i} possible input configurations for the function f_i that determines the value v_i of node a_i in the network. M_i^j represents the set of nodes which are inputs for f_i having taken their values in the jth input configuration out of all 2^{m_i} possible configurations. v_k^j represents the value of node $a_k \in M_i$ as dictated by M_i^j.

The probability of a particular input configuration arising at time $t-1$, as calculated by the inner binary expression, is added to the sum if the output of f_i for that configuration evaluates to 1. This is encoded by the Kronecker delta term in the outer summation, which takes the value 1 if $f_i(m_i^j) = 1$, otherwise takes the value 0.

The accuracy of $s_i(t)$ is evaluated by comparing it to the same probability determined via R simulations of the network with a synchronous update scheme from randomly selected starting states. Let this probability determined via simulation be $\bar{\sigma}_i(t)$.

$$\bar{\sigma}_i(t) = \frac{1}{R} \sum_{r=1}^{R} \sigma_i^{(r)}(t) \tag{4}$$

where $\sigma_i^{(r)}(t)$ denotes the value v_i of node a_i in the network at time t in the rth simulation.

The mean squared error and variance of probabilities determined via Equation 3 compared to Equation 4 are evaluated via the following expressions $e(t)$ and $b(t)$ respectively (which are described in more detail in [6]):

$$e(t) = \frac{1}{n} \sum_{i=1}^{n} [s_i(t) - \bar{\sigma}_i(t)]^2 \tag{5}$$

$$b(t) = \frac{1}{R \cdot N} \sum_{i=1}^{N} \sum_{r=1}^{R} \left[\sigma_i^{(r)}(t) - \frac{R\bar{\sigma}_i(t) - \sigma_i^{(r)}(t)}{R-1} \right]^2 \tag{6}$$

Applying this approach to the 40-node network described in [5], $e(t)$ and $b(t)$ were computed for various values of R and t. The results are displayed in Figure 16 and 17.

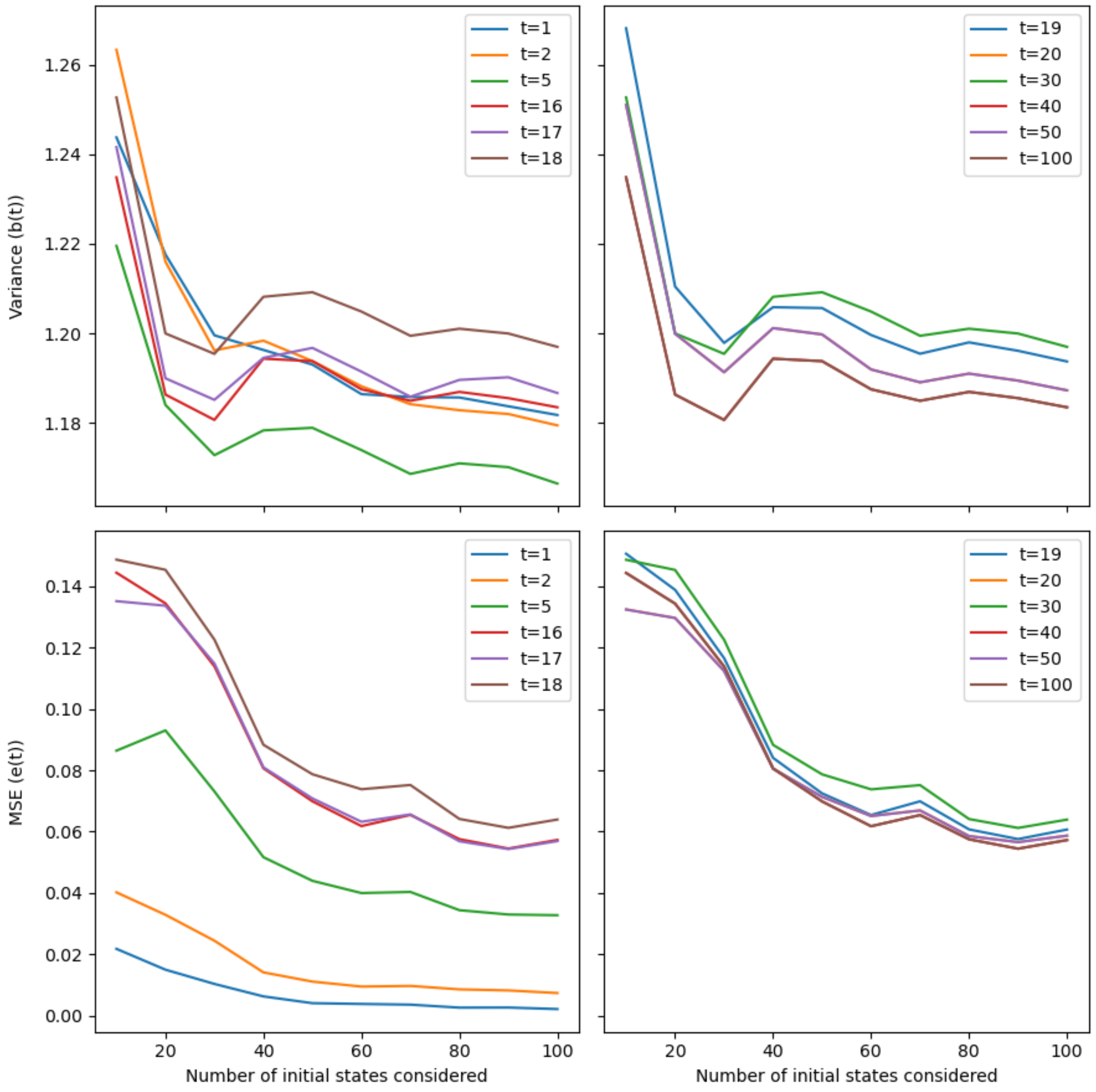

Figure 16: $e(t)$ and $b(t)$ over values of R upto 100 (i.e. initial number of states considered) for the 40-node network in [5], plotted at different times to identify the optimal number of initial states at which the MSE saturates.

Increasing R causes both $e(t)$ and $b(t)$ to start converging after a point. The values at which they plateau vary at different times t, as demonstrated in Figure 16. Though $e(t)$ does not tend to 0 as R and t increase, it is sufficient for it to attain a steady value, showing that the behaviour of the network demonstrated by the analytical approach and that obtained from the simulations have a constant 'separation' from each other above a certain R.

Figure 17 demonstrates that $e(t)$ and $b(t)$ attain saturation at about 6000 initial states being considered. Similar to the previous approach with the synchronous updates, if one could explicitly define a method to calculate a cutoff on the number of states beyond which there is negligible fluctuation in $e(t)$ and $b(t)$, this could help fulfil our aim of finding a subset of the network's state space that is sufficient to capture its dynamics through simulations.

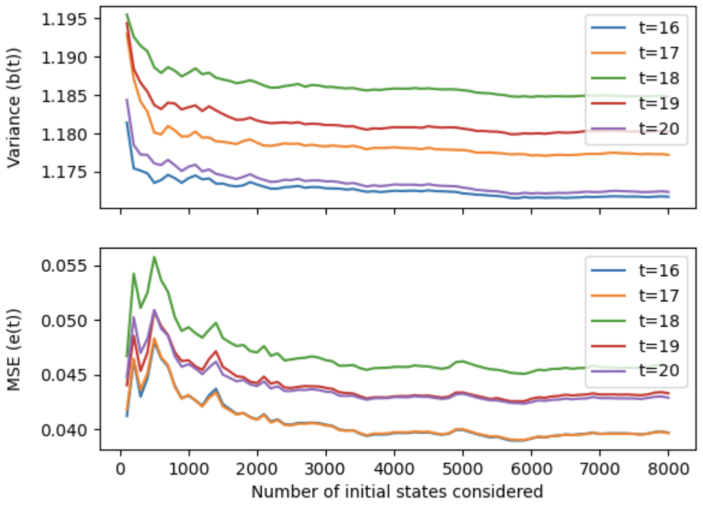

Figure 17: $e(t)$ and $b(t)$ over values of R upto 8000 (i.e. initial number of states considered) for the 40-node network in [5], plotted at different times to identify the optimal number of initial states at which the MSE saturates.

5 Conclusions and future work

Construction of a partial STG to approximate the dynamics of a large Boolean network poses two primary obstacles: limiting the number of permutations considered if an ROA update scheme is being used, and limiting the state space being considered. Both limits should be well-defined in such a way that the corresponding STG constructed allows accurate recapitulation of the network's behaviour.

Running the ROA update scheme to compute forward-transitions with a limited permutations q proves to be useful in estimating this behaviour, particularly in building a reasonably accurate approximation of the state-transition graph which can be used to make further inferences about the network (including calculations of the absorption probabilities of its transient states).

This approach lays the foundation for the BM-ProSPR algorithm, which introduces the Temporality ($\mathbb{T}^q$) and PageRank ($\mathbb{PR}^q$) metrics to deduce a suitable value of q. After optimising the time and memory complexity of important sub-routines, including those involved in the calculation of PageRank, the application of this algorithm returns a minimum number of permutations sufficient to construct a well-estimated partial STG.

The synchronous update scheme performed on an increasing subset of initial states of the network, and the individual-based mean-field approximation described in [6], present two possible approaches to estimate the minimum number of states sufficient for this purpose as well. Future work in this direction will thus involve defining cutoffs on the number of initial states used to calculate the steady-state probabilities or $e(t)$ and $b(t)$ respectively, and thus determining a reasonable limit on the state space of the network.

References

[1] Shubhank Sherekar and Ganesh A. Viswanathan. Boolean dynamic modeling of cancer signaling networks: Prognosis, progression, and therapeutics. *Computational and Systems Oncology*, 1(2), 2021.

[2] Assieh Saadatpour and Réka Albert. Boolean modeling of biological regulatory networks: A methodology tutorial. *Methods*, 62(1):3–12, 2013. Modeling Gene Expression.

[3] Shubhank Sherekar and Ganesh Viswanathan. Boolean dynamic modeling of tnfr1 signaling predicts a nested feedback loop regulating the apoptotic response at single-cell level. *bioRxiv*, 2022.

[4] S. Gupta, P.K. Panda, and W. Luo et al. etwork analysis reveals that the tumor suppressor lncrna gas5 acts as a double-edged sword in response to dna damage in gastric cancer. *Nat Sci Rep*, 12, 2022.

[5] S. Gupta, P.K. Panda, and R. F. Hashimoto et al. Dynamical modeling of mir-34a, mir-449a, and mir-16 reveals numerous ddr signaling pathways regulating senescence, autophagy, and apoptosis in hela cells. *Scientific Reports*, 12, 2022.

[6] T Parmer, LM Rocha, and F Radicchi. Influence maximization in boolean networks. *Nat Commun*, 13, 2022.