RESONCTRUCTING SIGNALING PATHWAYS FROM RNAI DATA USING GENETIC ALGORITHMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EYÜP SERDAR AYAZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
THE BIOINFORMATICS PROGRAM

JUNE 2011

Approval of the Graduate School of Informatics

_____

Professor Nazife Baykal

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Assistant Professor Didem Gökçay

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Associate Professor Tolga Can

Supervisor

Examining Committee Members

Associate Professor Ünal Erkan Mumcuoğlu	(METU, MIN)	_____
Associate Professor Tolga Can	(METU, CENG)	_____
Associate Professor Uğur Doğrusöz	(Bilkent CS)	_____
Assistant Professor Vilda Purutçuoğlu	(METU STAT)	_____
Assistant Professor Yeşim Aydın Son	(METU MIN)	_____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: **Eyüp Serdar Ayaz**

Signature: _____

ABSTRACT

RESONCTRUCTING SIGNALING PATHWAYS FROM RNAI DATA USING GENETIC ALGORITHMS

Ayaz, Eyüp Serdar

M.S., Bioinformatics Program

Supervisor: Associate Professor Tolga Can

June 2011, 41 pages

Cell signaling is a list of chemical reactions that are used for intercellular and intracellular communication. Signaling pathways denote these chemical reactions in a systematic manner. Today, many signaling pathways are constructed by several experimental methods. However there are still many communication skills of cells that are needed to be discovered. RNAi system allows us to see the phenotypes when some genes are removed from living cells. By observing these phenotypes, we can build signaling pathways. However it is costly in terms of time and space complexity. Furthermore, there are some interactions RNAi data cannot distinguish that results in many different signaling pathways all of which are consistent with the RNAi data. In this thesis, we combine genetic algorithms with some greedy approaches to find the topologies that fit the Boolean single knock-down RNAi experiments. Our algorithm finds nearly all of the results for small inputs in a few

minutes. It can also find a significant number of results for larger inputs. Then we eliminate isomorphic topologies from the output set of this algorithm. This process fairly reduces the number of topologies. Afterwards we offer a simple scheme for suggesting new wet-lab RNAi experiments which is necessary to have a complete approach to find the actual network. Also we describe a new activation and deactivation model for pathways when the activation of the phenotype after RNAi knock-downs are given as weighted variables. We adapt the first genetic algorithm approach to this model for directly finding the most possible network.

Keywords: signaling pathways, genetic algorithms, network construction, RNAi

ÖZ

SİNYAL YOLAKLARININ RNAI VERİLERİNDEN GENETİK ALGORİTMALAR KULLANILARAK YENİDEN OLUŞTURULMASI

Ayaz, Eyüp Serdar
Yüksek Lisans, Biyoinformatik Programı
Tez Yöneticisi: Doç. Dr. Tolga Can

Haziran 2011, 41 sayfa

Hücre sinyali hücre içi ve hücreler arası haberleşme için kullanılan bir dizi kimyasal tepkimedir. Sinyal yolakları bu kimyasal tepkimelerin sistemli bir şekilde ifade edilmesini belirtir. Günümüzde birçok sinyal yolağı değişik deneysel yordamlarla oluşturulmuş durumdadır. Lakin hücrelerin halen keşfedilmemiş birçok iletişim becerisi vardır. RNAi sistemi bazı genlerin yaşayan hücrelerden çıkarıldığında oluşan fenotipin görülmesini sağlar. Bu fenotipleri gözlemleyerek sinyal yolaklarını inşa edebiliriz. Fakat bu bu işlem zaman ve bellek büyüklüğü açısından maliyetlidir. Üstelik başı etkileşimler RNAi verileriyle keşfedilemez, zira birçok değişik yolak RNAi verileriyle uyumlu olabilir. Bu tezde ilk olarak genetik algoritmaları bazı sezgisel yaklaşımlarla birleştirerek tekli Boolean RNAi söndürme deneyleriyle uyumlu olan topolojilerin çoğunu bulduk. Algoritmamız küçük girdiler için neredeyse tüm geçerli sonuçları birkaç dakika içerisinde bulmaktadır. Ayrıca büyük girdiler için kayda değer sayıda geçerli sonuç üretmektedir. Sonrasındaysa bu

sonuçlar arasındaki izomorfik topolojileri eledik. Bu süreç topojilerin sayısını oldukça düşürdü. Sonuç olarak elimizde kalan topolojilerin tamamı eşit önceliktedir ve bunlar sınıflandırma algoritmalarının girdisi olarak kullanılabilir. Sonrasında asıl yolağı bulan eksiksiz bir sistem için yeni RNAi deneylerini yönlendiren bir sistem önerdik. Ayrıca RNAi deneyleri sonucundaki fenotipin aktifliğinin ağırlıklı değişkenler olarak verilmesi durumunda çalışacak bir aktivasyon modeli tasarladık. Bu model üzerinde en olası yolağı bulmak için ilk yaklaşımdaki genetik algoritmayı uyarladık.

Anahtar kelimeler: sinyal yolaklari, genetik algoritmalar, ağ oluşturulması, RNAi

ACKNOWLEDGEMENTS

Firstly, I want to thank my supervisor Tolga Can for his precious support during the preperation and writing processes of the thesis.

I also thank all of jury members. Their comments are really valuable.

I would like to thank my previous advisor Ali Aydın Selçuk for his lifetime support, especially for his encouragement to return to academic life.

I thank our PathoSys partner Professor Lars Kaderali for sharing the problem and the source code of their work with us.

This work is partially funded by the European Union in the 7th framework program through PathoSys, grant 260429.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Problem Definition and Motivation

Cell signaling is a system of chemical and physical reactions that provides cells to communicate with each other as well as sense the outer and inner cell conditions. It is necessary for all single and collective reaction of the cell such as survival, growth, and hormonal activities.

Signaling pathways denote the reactions between substrates that resolve how cell signaling proceeds. They are usually triggered by sensors or hormones and operated by proteins. There are several ways to schematize these pathways using experimental observations.

RNA interference (RNAi) technology is a recent research tool in systems biology that enables us to knock-down some genes in living cells. This is applied to observe typical phenotypes after systematically changing the genes of cells. By using these observations the genes can be correlated with some particular phenotypes. However building signaling pathways by using RNAi experiments is a relatively less examined topic.

As the current developments, lots of the signaling pathways are discovered. By using different methods, these pathways can be validated more confidently. Also

there are still many relations in the examined and non-examined pathways that are waiting to be recognized.

We have very few of the pathway parameters that are obtained by observing phenotypes after knocking-down genes one by one. Each experiment gives a single Boolean result which denotes that the phenotype, which indicates whether the pathway is active or not, is observed or not. This underdetermined situation cannot yield an exact signaling pathway. Instead, we can search the vast solution space to find most of the topologies that are consistent with single knock-down RNAi experiment data. In this thesis we use genetic algorithms to find a solution set. Then we propose a new system to direct new wet-lab experiments to find the actual pathway.

As the results of the first part, we significantly speed up the process of finding the feasible pathways.

The second part of the thesis deals with the same system of experiments but different inputs. Since the readout phenotypes are screened by their color or luminance as a result of the wet lab experiments, we can infer the activation rate of that phenotype as a real valued parameter. By using these activation rates of single knock-down RNAi experiments, we suggest a simple model to be used in a modified genetic algorithms approach that directly optimizes the network parameters to find the most suitable signaling pathway.

1.2 Related Work

Reconstructing signaling pathways is a research topic in which several methods are used. Markowetz *et al.*[1] introduced RNAi experiments to infer signaling pathways where expression profiling experiments do not work since they are modulated on non-transcriptional levels.

Moffat and Sabatini [2] take the advantage of using RNAi in high-throughput techniques to build mammalian signaling pathways.

Kaderali *et al.*[3] use signal knock-down data to observe the effects of different genes to a particular phenotype. To overcome the lack of necessary input data, they design a Bayesian learning approach in networks with probabilistic Boolean threshold functions. They also employ Metropolis-Hastings algorithm to sample network variables. Their approach can be improved with inferring the distributions over the topologies that are consistent with the experiments.

Similarly, using RNAi knock-down experiments, Ourfali *et al.* [4] define a framework called SPINE (Signaling-regulatory Pathway INferencE) that uses both protein-protein interaction networks and protein-DNA interaction networks and combine them in a new modified graph to search for new interactions between genes.

Zhu *et al.* [5] use both biological data sources such as genetic epitasis analysis and statistical approaches to discover and order the pathway components with a *de novo* signaling pathway reconstruction method.

Recently, Acharya *et al.* [6] proposed a novel framework which consists of two stages. In the first stage, they use molecular profiling data to find overlapping linear signal transduction events which they call as Information Flow Gene Sets. In the second stage, they design a Gibbs Sampling based algorithm that is used to build pathways on the gene sets found in the first stage.

For a more comprehensive review of network construction processes, a very helpful source can be found in Markowetz *et al.* [7].

1.3 Contributions

The contributions of this thesis to the signaling pathway reconstruction literature are:

1. We propose a novel method which uses genetic algorithms for reconstructing signaling pathways. This method finds almost all of the

correct topologies for small inputs in a small amount of time. For large inputs, it finds reasonably many distinct solutions in a few minutes.

2. We propose a greedy approach to the GI-complete directed acyclic graph isomorphism problem which uses longest paths from a source node to a sink node.
3. We make an observation on the topologies that is used to divide the problem into smaller versions of the same problem. Also we find a dynamic programming method to count the number of distinct topologies.
4. We also define a new model and propose another genetic algorithm approach for weighted readout input version of the same problem.

CHAPTER 2

Theoretical Basics and Biological Background

2.1 Signaling Pathways

The fundamental characteristics of living organisms are response to stimuli, development, reproduction, energy usage, chemical and physical organization and adaptation. All of these have a chemical background beyond the interface of observed actions. Upon the chemistry, cell signaling is the trigger that directs these actions. It is necessary for communication of cells as well as sensing the extracellular and intracellular conditions.

Cell signaling can be initiated by three different types of signals. Some signals can be transmitted by direct contact of cells, which is called juxtacrine signaling [8]. Some of the signals do not need touching of cells but can be transmitted in short distances like neurotransmitters. This type of signaling is called paracrine signaling [8]. The other signaling type is endocrine signaling which can be transmitted from distant parts of the body via endocrine system [8]. The transmitters of these signals are hormones.

Signaling pathways are the chemical back side of the cell signaling. They take part when the signals reach the cell. They are perceived by special proteins called receptors. By sensing these intracellular and extracellular signals a series of complex chemical reactions is triggered resulting with the response of the cell. This complex

system is operated by proteins. With the recent technological developments we are able to discover all the reactions and ligands that take part in the process. The final schema of these subunits and reactions is called signal transduction pathways or shortly signaling pathways. In Figure 1.1, a general view of signaling pathways is shown.

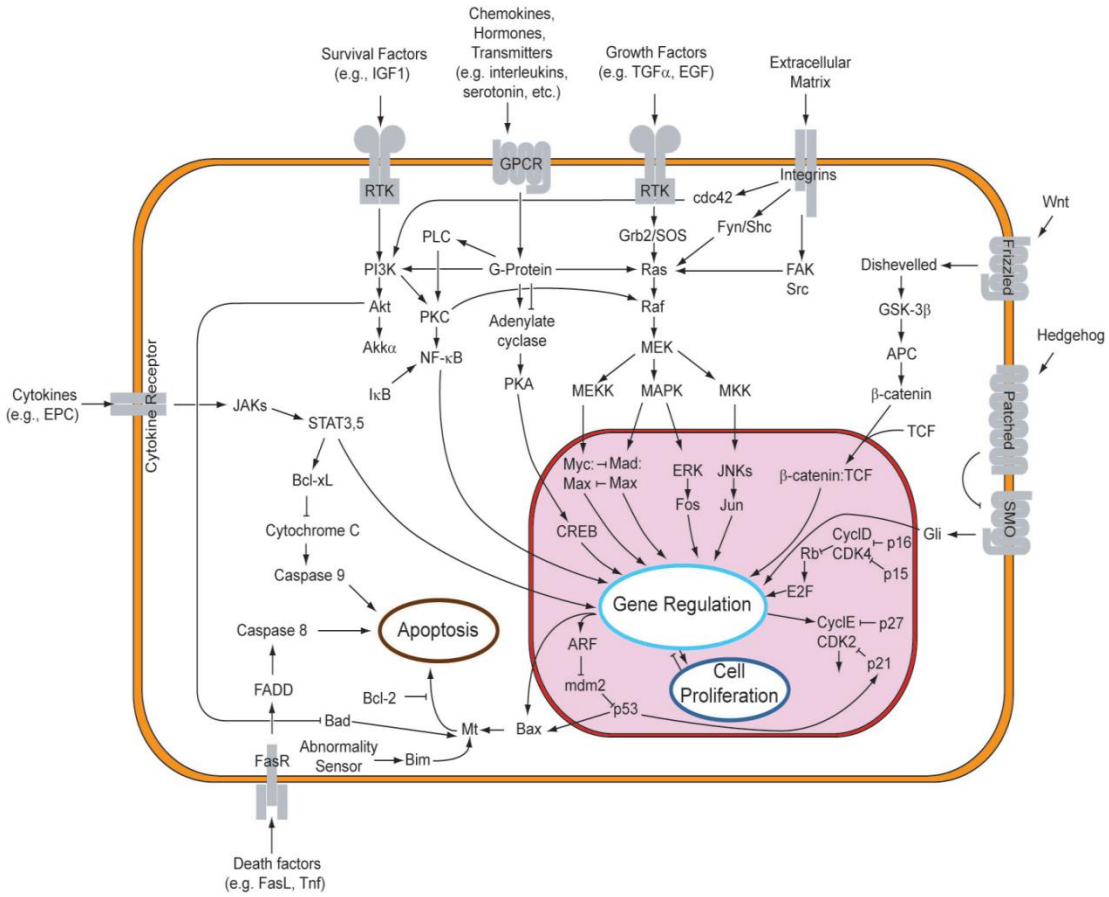


Figure 2.1: A general view of signaling pathways [9].

As an analogy between computer systems and biological systems, we can match cell signaling with prompting executable programs. They can be invoked by computer user or another process. Signaling pathways match with the executable programs where the chemical reactions are single instructions. Each instruction is succeeded by another instruction that uses the data created by previous instructions.

2.2 RNAi Technology

RNA interference technology is a system that enables us to change the activation level of genes in living cells [10]. Although it can be used for upregulation of genes, it is usually utilized for silencing a particular gene, which we call as knock-down. Gene knock-down is a robust method used in different biotechnological applications.

The pathway of knock-down mechanism is as follows. Firstly, RNA is synthesized to silence a particular gene and this synthetic RNA called shRNA is inserted into a phage virus. By using this phage, shRNA is transported to a living cell's cytoplasm. In the cytoplasm, shRNA is transcribed into its reverse strand with reverse transcriptase enzyme and then it passes into nucleus to be injected into DNA. Then this part of DNA is transcribed into pro shRNA which has a hairpin shape. Pro shRNA is fragmented into 20-25 base pairs double-stranded RNA by a protein called Dicer. These double-stranded RNAs are called small interfering RNA (siRNA). After that, an siRNA is unwound and one of the single strands is incorporated into the RNA-induced silencing complex (RISC). RISCs bind to mRNAs and prevent them to be translated into proteins. This process is called post-transcriptional silencing. Also an siRNA can incorporate into a different complex called RNA-induced transcriptional silencing (RITS) that binds to DNA to break-up transcription. The mechanism of RISC and RITS can be seen in Figure 1.2.

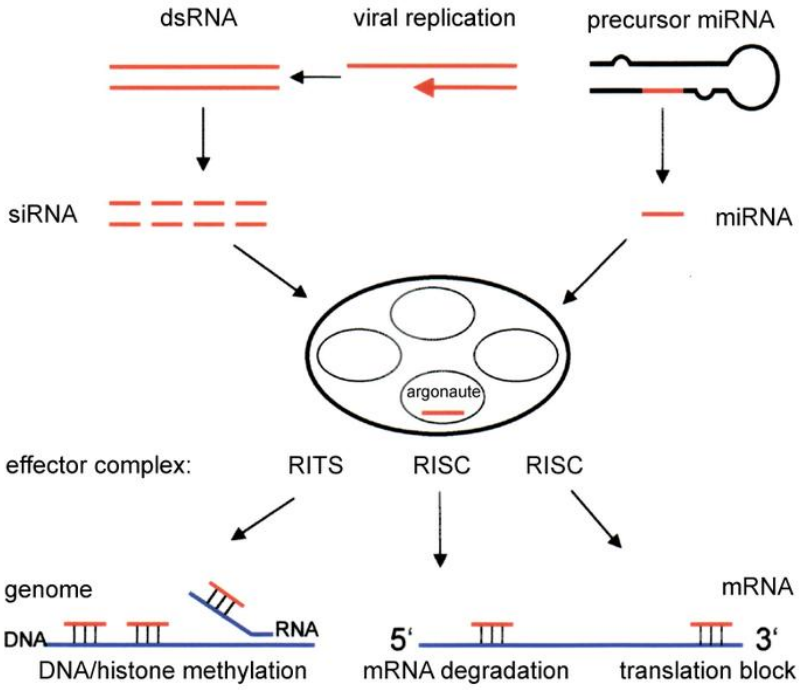
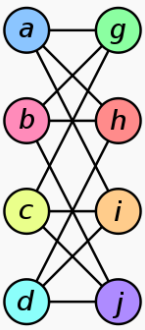
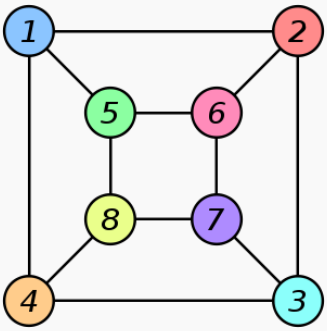


Figure 2.2: The mechanism of RITS and RISC [11].

2.3 Graph Isomorphism

Definition: Let $G(V_G, E_G)$ and $H(V_H, E_H)$ are two graphs. A one to one and onto function $f: V_G \rightarrow V_H$ defines an isomorphism when the following proposition is satisfied: $(a, b) \epsilon E_G \Leftrightarrow (f(a), f(b)) \epsilon E_H$.

Table 2.1: A graph isomorphism example [12].

Graph G	Graph H	An isomorphism between G and H
		$f(a) = 1$ $f(b) = 6$ $f(c) = 8$ $f(d) = 3$ $f(g) = 5$ $f(h) = 2$ $f(i) = 4$ $f(j) = 7$

Graph isomorphism problem is the determination of whether two definite graphs are isomorphic or not. This problem is definitely in NP but it is not proven yet that it is in NP-complete or P. A similar problem, subgraph isomorphism problem, is determining whether there exists a subgraph of a definite graph which is isomorphic to another definite graph. Subgraph isomorphism problem is proven to be NP-complete. If graph isomorphism problem is in NP-complete, than polynomial time hierarchy will collapse, so it is predicted not to be in NP-complete. A new complexity set is defined for this problem called GI-complete (Graph Isomorphism-complete).

Directed acyclic graph (DAG) isomorphism is also in GI-complete. A polynomial time reduction from graph isomorphism to DAG isomorphism is as follows: For a graph *G*, create a new directed graph, *H*, that has a vertex for each vertex and edge in *G*. For each edge *(a, b)*ϵ*G*, add two directed edges to *H*, from the vertices of *H* that correspond to the vertices of *G*, *a, b*, to the vertex of *H* that correspond to the edge in *G (a, b)*. The directed graph of *H* is acyclic [13].

2.4 Genetic Algorithms

Genetic algorithms are first developed in mid-70's as an inspiration from Darwin's Theory of Evolution [14]. Although their original purpose of development is to transcribe the concept of natural adaptation mechanism to artificial systems, they have been used to solve optimization problems in very complex models.

Generally, genetic algorithms have these requirements [15]:

- All possible solutions should be represented as a data set called chromosomes.

- A very fast and accurate evaluation function that returns a comparable value for all possible solutions. The proportion of the evaluation value over the average evaluation value of the pool is called the fitness function.

- A set of modification functions. They are usually classified in two groups: Mutations and cross-overs. Mutations are rare and make small changes on chromosomes that are chosen randomly. Cross-over is the exchange of genes from two chromosomes. It is more frequent than mutations.

- A selection mechanism over the chromosome pool that favors the better chromosomes to survive and the worse ones to be swept out.

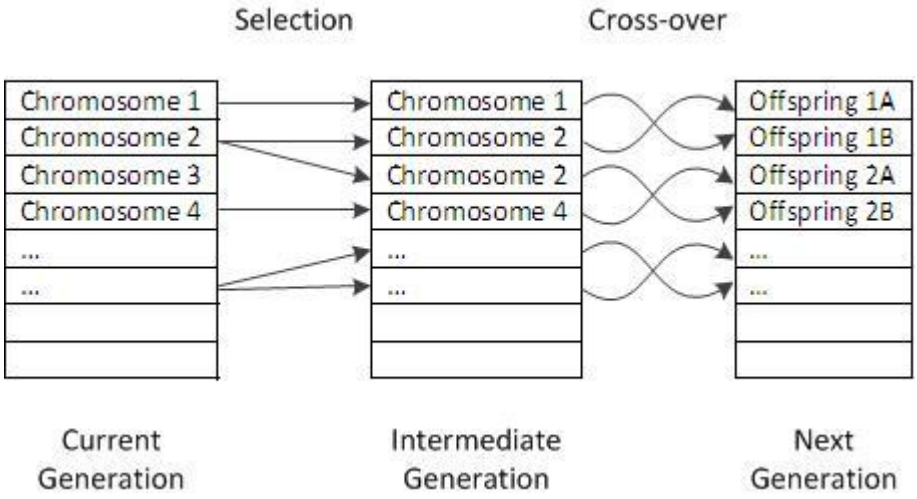


Figure 2.3: One generation of a genetic algorithm.

Upon these features, genetic algorithms work as follows. First, a large pool of chromosomes is created randomly and every chromosome is evaluated. The chromosomes can be chosen from any distribution but a sampling that represents the actual population of solutions will perform more efficiently. The initial pool is accepted as the first generation and a process starts. For each generation, first a selection phase occurs to create an intermediate generation. In the selection mechanism, a chromosome is expected to be represented by a number which shows its fitness. Couples of chromosomes are mated from the intermediate generation and they are crossed-over. Then mutations occur with a very small probability for each gene. This process yields a new generation. The whole process is iterated until a convergence state or for an amount of time. A view of selection and cross-over is shown in Figure 1.3.

CHAPTER 3

Signaling Pathway Reconstruction Processes

3.1 Boolean Data

In this section, we attack the signaling pathway reconstruction problem where the observation of phenotypes after RNAi single knock-downs are given as Boolean variables

3.1.1 Problem Definition

The original problem is described in detail in the supplementary material of Kaderali *et al.* [3]. In this problem observing just one phenotype in terms of a marker gene after stimulating a gene is the initial situation in which the phenotype should be in active state. By using the RNAi system, we can knock-down some genes in a cell. For availability reasons, the RNAi data are restricted to single knock-downs and only activation interactions are considered. The state of the phenotype after knocking-down each regulatory node, except the stimulated node and the observed node, one by one is given as the input. This input data are not sufficient to yield a unique network topology. For example, according to Figure 3.1, both (a) and (b) are consistent with the single knock-down data given in the table. By using the combinatorial knock-down data given in the table, we can see that only (a) satisfies the conditions, but it is impossible to identify that in our case.

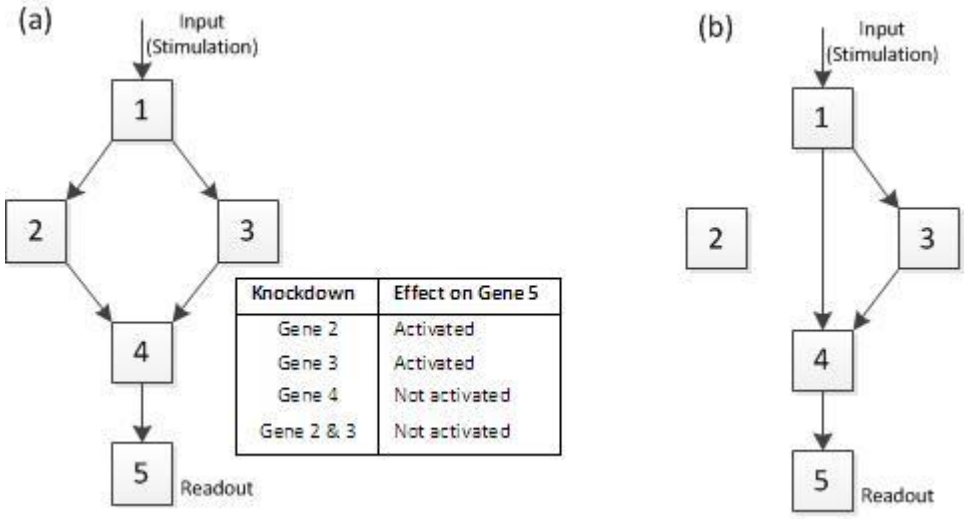


Figure 3.1: Knock-downs and effects

3.1.2 Restricted Cases

Also some topologies are restricted because of identifiability reasons. The topologies that are unidentifiable in this condition set are written and illustrated as follows:

- Upstream nodes of the stimulated node and downstream of the screening node (Figure 3.2(a))
- Isomorphic graphs ((Figure 3.2(b)))
- Feedback loops (Figure 3.2(c)) and feedforward loops (Figure 3.2(d))
- Unreachable nodes from the stimulated node and the nodes that cannot reach the screening node (Figure 3.2(e))

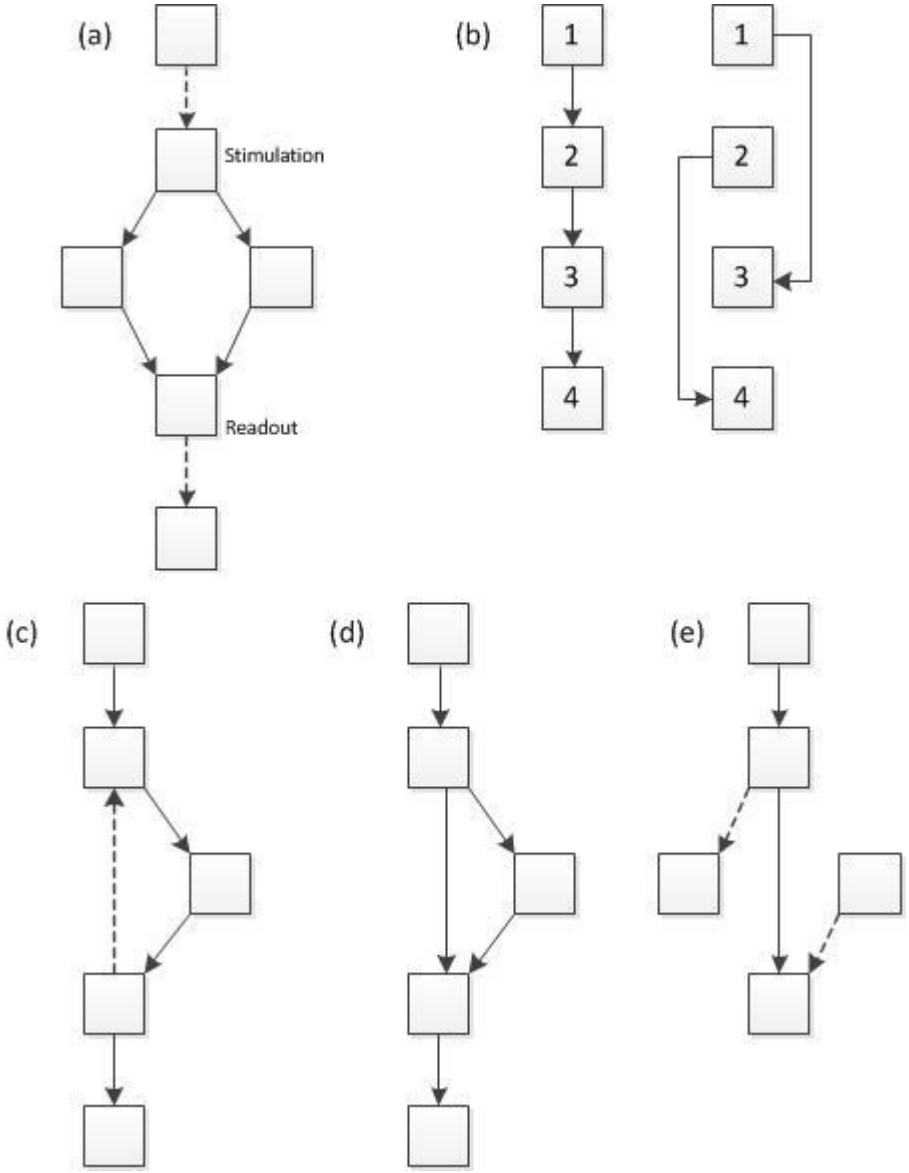


Figure 3.2: Restricted topologies

3.1.3 Problem Complexity

With all the limitations listed in the previous section, the problem is to construct almost all of the network topologies consistent with single knock-down data. The solution space is very large. Since there are $n(n\text{-}1)$ possible interactions in a network of n nodes, the solutions space is $2^{n(n-1)}$. On the other hand the input variables are only the single Boolean phenotypes after knocking-down each gene one by one. Moreover inputs can be reduced to the total number of nodes and the number of nodes that do not have an effect on the activation of the readout node when it is

knocked down, m, where $0 \leq m < n-2$. Because the order of the bits in Boolean phenotypes does not give more information than the number of 0s in that list of bits. As shown in the Figure 3.3, the input sets (n=8, phenotype=001001)(a) and (n=8, phenotype=100100)(b) give the same information, since the topology that is consistent with the first input can be applied to the second input by just changing the order of node numbers.

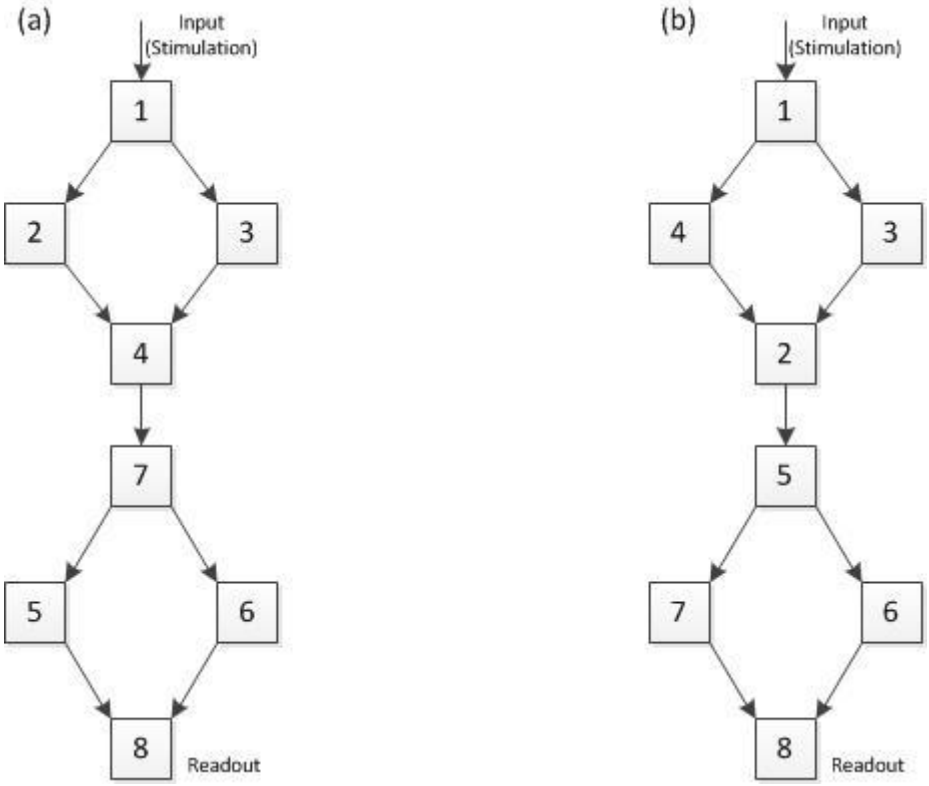


Figure 3.3: Different phenotypes, same topologies

3.1.4 Genetic Algorithm Approach

Unlike the common usage of genetic algorithms, our problem is not an optimization problem. Yet, genetic algorithms are suitable for our problem, because we do not have to find all of the topologies that are consistent with the input data. Our approach starts with lots of random points and tries to adapt them to the input conditions gradually.

We started to design our algorithm by determining the chromosome representation. Obviously, we have chosen the adjacency matrix as the chromosome. Since the upstream nodes of the source node and downstream nodes of the sink node are restricted, the first column and the last row of the adjacency matrix is always 0. Thus we can use *(n-1)*x*(n-1)* matrices. Also we can use topologically sorted networks since they are directed acyclic graphs. Only forward edges on a topologically sorted graph can exist, so the lower part of the diagonal of adjacency matrix is always 0.

The initiation of first generation can be done with several ways. On a topologically sorted graph, the upper side of the diagonal of an adjacency matrix can be assigned to 0 and 1 randomly. However it will yield a highly dense graph which always stays connected after single knock-downs. Even if the probability of 0 is assigned higher, the edges from low ordered nodes to high ordered nodes will make the nodes between them ineffective. For that reason, we decided to add an edge from node *i* to mode *j* where $1 \leq i < j \leq n$, with a probability of p^{j-i}. The value of p which makes the expected value of the number of effecting genes m is shown in Equation 1. But it is hard to solve this equality, so we make some simplifications on the equation and take the value of p as in the Equation 2. If the indegree of a node except the source is 0, then we add a random edge that enters to that node. Similarly we do this process to the nodes which has 0 outdegree except the sink node. After that we yield a graph in which every node can be accessible from the source and every node has a path to access the sink.

$$m = \sum_{k=2}^{n-1} \prod_{i=1}^{k-1} \prod_{j=k+1}^{n} 1 - p^{j-i}$$

Equation 1

$$p = 1 - \frac{1}{1 + \sqrt{\frac{n-2-m}{n-2}}}$$

Equation 2

Our modification functions have a small degree of some consciousness. Cross-over function is simply exchange of the rows in the adjacency matrices after a cutting point. This is called as 1-point cross-over in genetic algorithm terminology [15]. Also the cross-over points can be chosen as articulation points, but we do not prefer this strategy, because it will reduce the available cross-over points. Also it may cause premature convergence of the pool. The mutation function works similarly. Some entries are flipped in the matrix in order to make break convergence of some points in the graph. After doing modifications, we may have some intermediate sources or sinks. They are removed by adding incoming edges to the sources and outgoing edges to the sinks. Other features of the graph are not exceeded by these modifications.

Our evaluation function finds the total number of articulation points (a). Articulation point represents the nodes which makes the graph disconnected when they are taken out. Since all of the nodes are reachable from the source and reach the sink, a is the exact number of nodes that affect the screening phenotype after knocking-down. The evaluation function is shown in Equation 3. This equation takes values in the interval [0, 1] and higher value represents better results. We use square root function to amplify the error rates. The evaluation function works in $O(n^2)$ time which is admissibly fast since n is not so high.

$$evaluation(a) = 1 - \sqrt{\frac{|m-a|}{\max(m, n-2-m)}}$$

Equation 3

Algorithm 1 Articulation Point Finding

Input: N, adjacency[N][N]
Output: articulation_points
reach ← 0
articulation_points ← ∅
for i=1 **to** N-1 **do**
    if i>1 **and** reach≤i **then**
        reach ← i
        articulation_points ← articulation_points ∪ i

```
        for j=i+1 to N do
            if adjacency[i][j] and j>reach then
                reach ← j
```

The selection mechanism when advancing generations is different than canonical genetic algorithms method. In our algorithm, the chromosomes are selected in favor of their evaluation functions over average evaluation function, and after modification functions are applied to them, they are thrown to the intermediate pool. After finishing the modification phase, the matching chromosomes, i.e. chromosomes with an evaluation value of 1, are removed from the intermediate pool. They are written in a file if they are free of feedforward loops. The open slots of the intermediate pool are filled by randomly chosen elements from the original pool and our intermediate generation becomes the next generation. Therefore a chromosome can remain in the pool without modification, unlike the canonical genetic algorithm. Actually this approach is called as $(\mu+\lambda)$-ES (Evolution Strategy) in genetic algorithms society [16]. Searching the graph for feedforward loops requires all-pairs longest path algorithm in a directed acyclic graph which works in $O(n^3)$ time. The graph is free of feedforward loops if there is not an edge between two nodes when there exists a path of length more than one between these nodes.

Algorithm 2 All Pairs Longest Path

```
Input: N, adjacency[N][N]
Output: longest[N][N]
Set all elements of longest 0
for i=1 to N-1 do
    for j=i+1 to N do
        if adjacency[i][j] then
            for k=1 to i do
                if longest[k][i]>0 or i==k then
                    if longest[k][i]+1>longest[k][j] then
                        longest[k][j] ← longest[k][i]+1
```

Our algorithm continues unless the pool converges to a small group of chromosomes or it iterates for a definite number of generations. At the end, we have a file in which the solutions are written. In this file there may be duplicates. The

graphs are reorganized and sorted to put the duplicates together. All the limitations except isomorphism are satisfied in the resulting list.

3.1.5 A Greedy DAG Isomorphism Elimination

Using topologically sorted graphs permits some of the isomorphic topologies. An example for this can be seen in Figure 3.4(a). However there are still isomorphic graphs that can be represented by different adjacency matrices. For example Figure 3.4 (b), (c) and (d) have the same topology but different adjacency matrices. For this reason, we reorganized the adjacency matrices to eliminate isomorphic cases.

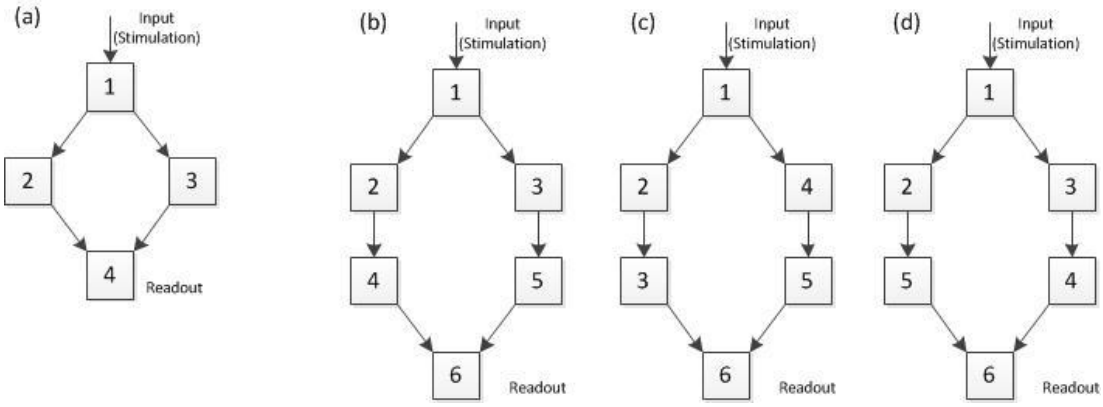


Figure 3.4: Topologically sorted isomorphic graphs

Our problem is a special case of DAG isomorphism. In this problem there is just one source node and one sink node. Yet it is still in the GI-complete class. A directed acyclic graph can easily be reduced to our case by just adding one source node, one sink node and edges from the newly created source node to the nodes of which indegree is 0 and from the nodes of which outdegree is 0 to the newly created sink.

Using GI-complete isomorphism checks would be too slow, since there are billions of results to be compared. Therefore we decide to reorganize the adjacency matrices and give them a lexicographical order. For that, instead of using previously developed algorithms, we develop a simple and fast sorting scheme for reorganization.

We define the concept of primary level of nodes, which indicates the longest distance from the source node. Firstly, we sort the nodes by their primary levels in ascendant order. Of those which have equal primary levels are sorted descendent by a second level which is the longest distance from the node to the sink node. If both of these levels are equal, the one which has the lower numbered descendent takes precedence and this criteria is computed from source to sink, like an insertion sort, in order to avoid ambiguities. At last, if all criteria are still identical, indegrees and outdegrees of the nodes are considered in sorting.

The longest distances have already been calculated when feedforward loops are checked, so it would not cost any more time for us. Just two simple sorts are enough for this reorganization. The levels of articulation points are unique and the reorganization does not violate topological sort. Because of all these reasons we choose to use this approach to eliminate most of the isomorphic topologies.

3.1.6 Further Observations

Divisibility of the Graph

For a topologically sorted directed acyclic graph, all nodes between an articulation point and the source can reach the articulation point as well as all nodes between the articulation point to the sink node can be reached from the articulation point. This means that an articulation point can divide the graph into two graphs, both of which share only that articulation point. These two graphs have the same properties and satisfy the same limitation with the original graph. So we can divide the graph into the atomic subgraphs that have no articulation points. An example can be seen in Figure 3.5. The topology in (a) is divided into two topologies in (b).

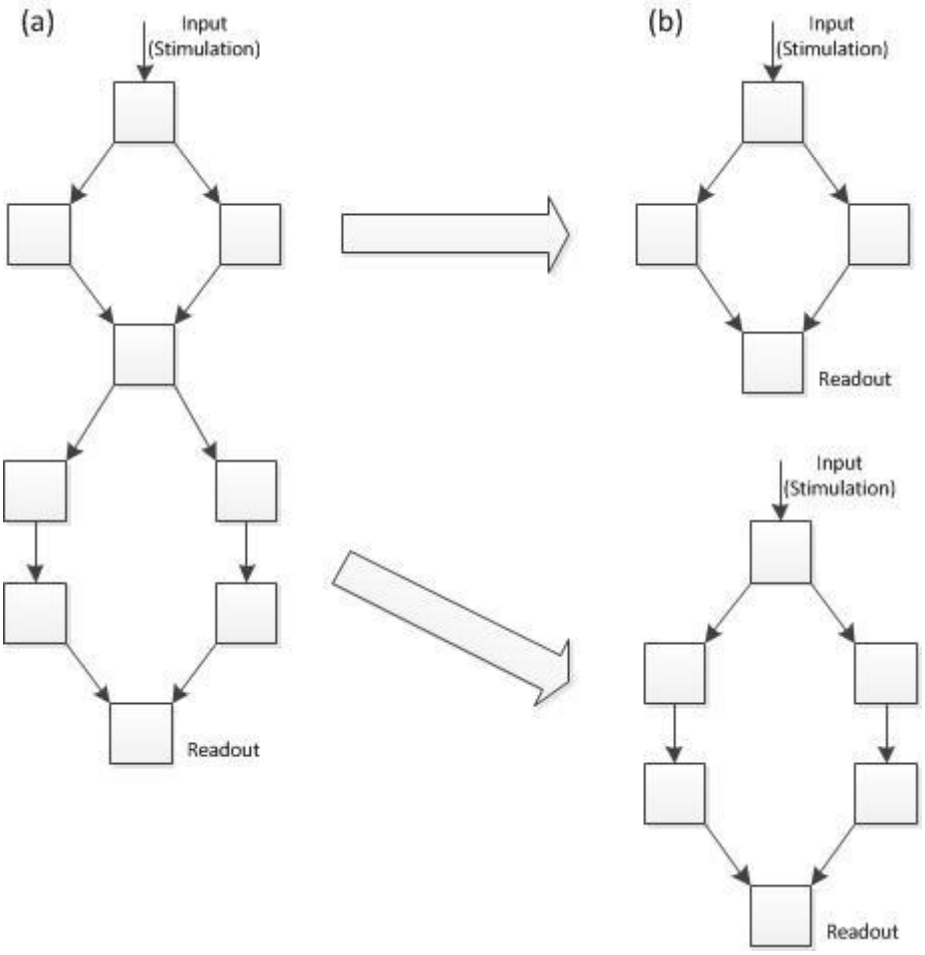


Figure 3.5: Division of a graphs

By using this property the problem can be divided into smaller problems each of with zero articulation points. Then they can be merged easily.

Dynamic Programming for Counting Topologies

Divisibility of the graph can be applied to count the number of solutions where the numbers of atomic graphs are known. Let *C(n, m)* be the number of distinct solutions where *n* denotes the number of nodes and *m* denotes the number of articulation points. Then the recurrence relation is formulated as where *m>0*:

$$C(n,m) = \sum_{i=2}^{n-m} C(i,0)C(n-i+1,m-1)$$

The $O(n^2m)$ solution when the first column is known:

Algorithm 3 Counting Topologies

Input: N, M, first column of C
Output: C[N][M]
Set all elements of C except first column 0
for j=1 **to** M **do**
    for k=j+2 **to** N **do**
        for i=2 **to** k-j **do**
            C(k,j) ← C(k,j)+C(i,0)*C(k-i+1,j-1)

Reverse Networks

For a solution, changing the direction of edges and exchanging the source and sink nodes is also a solution as seen in Figure 3.6. Finding a solution, unless it is symmetric, gives us another solution. Furthermore this property can be applied to each atomic part of the solution. Therefore we can extract up to 2^{m+1} different solutions from one solution.

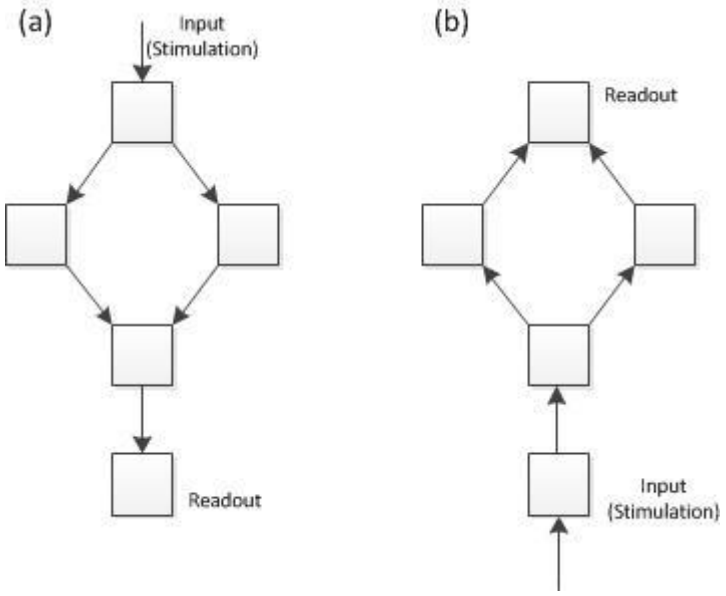


Figure 3.6: A reversed topology

3.1.7 Suggesting New Wet-Lab Experiments

Up to this point, we just search for topologies, not exact pathways with definite labels on nodes. With these Boolean single knock-down data, we cannot find any

more. However this data can lead us to make new experiments. Considering time and financial costs of wet-lab experiments, starting with single knock-down RNAi screens is a reasonable act.

Changing the readout node or the stimulation node can be used as a compare function between articulation points: Let x and y be two distinct articulation points. When we stimulate x and knock-down y; if the actual readout node is activated then y is a predecessor of x, else y is a successor of x. Similarly, after the stimulation of the original root and knock-down y, if x can be set as a readout node, we can determine the order of x and y in the network.

We need to make at least $O(mlogm)$ interactive experiments to sort the articulation points. If we sort it by merge sort, we can parallelize it. If an experiment takes t time, then the sorting process finishes in $\theta(tlogm)$ time in $m/2$ parallel ways.

After determining the order of articulation points, same comparison process can be applied to non-articulation point nodes. The narrowest interval, which means the node set that are placed between two articulation points, of a node can be determined by binary search over the articulation points so this process needs $\theta(nlogm)$ time. A nodes place do not depend on the other experiments' results so all $n-m$ nodes can be processed in parallel. This process takes $\theta(tlogm)$ time.

These are all we can find with single knock-down Boolean data. After this point double knock-downs can be applied among the non-articulation points of the same interval.

3.2 Weighted Data

In this section, we suggest a new model for the signaling pathway reconstruction problem where the observation of phenotypes after RNAi single knock-downs are given as weighted variables. Then we adapt the proposed genetic algorithm approach for Boolean data by slightly changing the algorithm details and some of the variables.

3.2.1 Modeling

Activation of the genes is determined by a florescent label in the experiments. These florescent labels can be measured by their color or the luminance to define the activation rates of genes. In the previous section, these activation rates are converted to Boolean values. But using the real values of these analog signals would give us more information. This information is so valuable, since with the Boolean variables the problem remains underdetermined. We design a model in which the activation rates can be any real number from 0 to 1. Higher values denote more active genes.

There are several probabilistic and graph based models to represent signaling pathways described in Markowetz *et al*[7]. However in the context of this thesis we are using genetic algorithms which require very fast and simple models. For that reason we cannot use any of the previously described methods, therefore we defined our own simple method.

In this model we have one variable for initial activation rate of the read-out gene in addition to n-2 activation rates for knock-downs. They are denoted as *observation(y)* where y is the knocked-down gene. Initial activation rate is given when $y=0$. In the graph, edges take values between -1 and 1. Negative valued edges represent deactivation interaction.

Our graph is directed and acyclic. But unlike the Boolean model, there can be feedforward loops in this one because they can be distinguished now. Except feedforward loops, all other limitations are present in this model.

In our model, all nodes have activation rates between 0 and 1. We use *activation(v,y)* function to denote the proposed activation rate of node v after knocking-down node y. If $y=0$ then this means none of the nodes is knocked-down. The stimulation node's activation rate is always 1 and the knocked-down node's activation rate is 0. For the other nodes in our graph G, activation value of node v is calculated as

$$activation(v, y) = \sum_{(u,v)\in G} activation(u, y).weight(u, v)$$

Equation 4

If the activation value of a node is less than 0 or greater than 1, then it is converted to 0 or 1.

3.2.2 Genetic Algorithm Approach

Boolean network problem is not an optimization problem but the problem defined in 3.2.1 is. We are searching for the most suitable weighted directed acyclic graph, so we can look for an optimization over an evaluation function. Our approach starts with lots of random points and tries to improve the average evaluation of pool generation by generation.

Similar to the Boolean approach, we use adjacency matrices as chromosomes. In the Boolean problem there are just two types of nodes. The challenge of this problem is that all the nodes are distinct now. Therefore we cannot use topologically sorted over the diagonal matrix representation with the original node numbers. Instead, we use another mapping function from nodes to the topological sort orders in addition to the topologically sorted adjacency matrix.

We start the initiation of first generation with a permutation of nodes except the stimulation and readout genes which always take the first and the n^{th} places. We define the number of genes that affect the readout as:

$$m = \sum_{i=2}^{n-1} |observation(0) - observation(i)|$$

Equation 5

This number is also compatible with the Boolean approach. We do not want our graph to be unnecessarily dense. Hence we decide to use the same probability p^{j-i} for adding an edge from node i to mode j where $1 \le i < j \le n$ and taking p as in the Equation

2. But this time the edges are weighted and deactivations are possible. When we are assigning the weights, we consider the changes of the phenotype. If both *activation(n,0)-activation(n,i)* and *activation(n,0)-activation(n,j)* are positive or negative then *weight(i, j)* is assigned positive, otherwise it is assigned negative. If we knew the activation of all nodes we can define the correlation of the two nodes by Pearson's correlation, but we know only the read-out node. So we choose *weight(i, j)* in a normal distributed scheme where the mean and standard deviation are:

$$\mu = (activation(n,0) - activation(n,i))(activation(n,0) - activation(n,j))$$

Equation 6

$$\sigma = \frac{\sqrt{(activation(n,0) - activation(n,i))^2 + (activation(n,0) - activation(n,j))^2}}{2}$$

Equation 7

As the Boolean problem if the indegree or outdegree of a node except the source and the sink is 0, then we add random edges to make sure that every node can be accessible from the source and every node has a path to access the sink.

The most challenging part of the weighted variables approach is the cross-over function. We cannot directly translate the 1-point cross-over method to this one, since the up and down cross-over point node sets of two chromosomes would not match. Because of that we choose n-point cross-over method[15]. In this method, just one row of the adjacency matrix exchanges with the corresponding row of the other adjacency matrix and then they are adapted to their new matrices. The topological sorts are not changed at all.

There are more than one mutation function. One of them makes random swaps on the topological sorts. This mutation is essential since it is the only way to change initial topological sorts. The other mutation adds, removes edges or changes the weight of an edge. After modifications, intermediate sources or sinks are removed by adding incoming edges to the sources and outgoing edges to the sinks.

The evaluation function is negatively correlated with the (*n-1)*-dimensional Euclidean distance between the proposed activation rates of the screening node and the actual input. Our evaluation function is in [0,1] interval where higher value means more suitable solution. Again, we used square root to amplify error. The function is shown in Equation 9. Finding the proposed activation rates is the bottleneck of the evaluation function which takes $O(n^3)$ time.

$$euclidean\_distance = \sqrt{(activation(n,0) - observation(0))^2 + \sum_{i=2}^{n-1}(activation(n,i) - observation(i))^2}$$

Equation 8

$$evaluation = 1 - \sqrt{\frac{euclidean\_distance}{\sqrt{n-1}}}$$

Equation 9

Algorithm 4 Proposing Activations

```
Input: N, weight[N][N], knocked_down
Output: activation[N]
Set all activation values 0
activation[1] ← 1
for i=2 to N do
    for j=1 to i-1 do
        if j≠knocked_down then
            activation[i] ←activation[i]+activation[j]*weight[j][i]
    if activation[i]<0 then
        activation[i] ←0
    if activation[i]>1 then
        activation[i] ←1
```

The selection mechanism puts the current generation and intermediate generation to the pool, sorts them by their evaluation function and inherits the top half of the pool to the next generation. This is also a (μ+λ)-evolution strategy [16]. The chromosomes of which evaluation function is 1 are not removed from the pool unlike the Boolean model. This means the actual solution is found, so the algorithm stops and returns

that solution. Actually it is not very probable to find the perfect solution without simplifications or limitations.

Our algorithm continues unless the pool is not being improved for a while or it iterates for a definite number of generations. A threshold evaluation value is specified at the end and during the iterations, above the rim solutions are written in a file. There solutions can also be discretized.

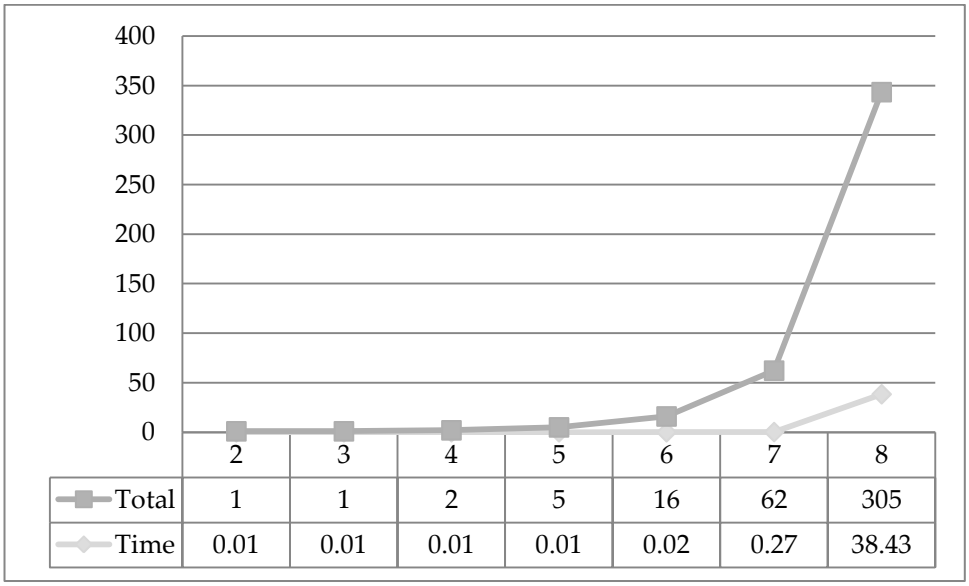
CHAPTER 4

Experimental Results

In this section we present the experimental results of our algorithms. We implemented the algorithms in C language. All the computations below were run on a laptop having 1.66 GHz processor and 1.5 gigabyte memory.

4.1 Exhaustive Search

We first implement exhaustive search algorithms for comparing the results with our algorithms.



	2	3	4	5	6	7	8
Total	1	1	2	5	16	62	305
Time	0.01	0.01	0.01	0.01	0.02	0.27	38.43

Figure 4.1: Growth of time and the number of solutions

As seen in the Figure 4.1, the growth rate of both the number of solutions and the execution time is drastically. Actually the rate is factorial which is even greater than exponential.

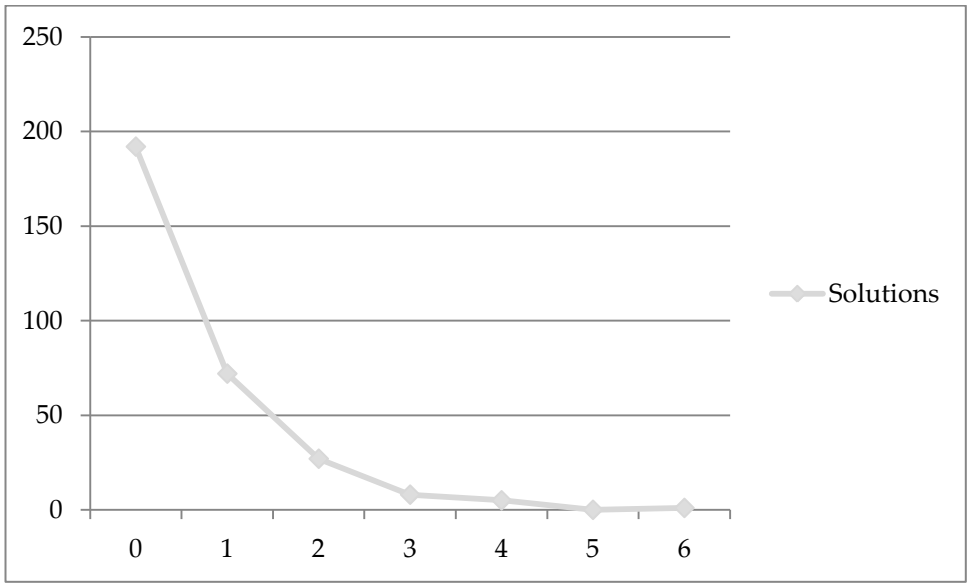


Figure 4.2: The number of solutions for n=8

Figure 4.2 shows that when m increases, the number of solutions decreases. Since the graphs with lower m values are usually denser, there are more distinct results for them. Also the percentage of the solutions of smaller m values increase with higher n as seen in Figure 4.3.

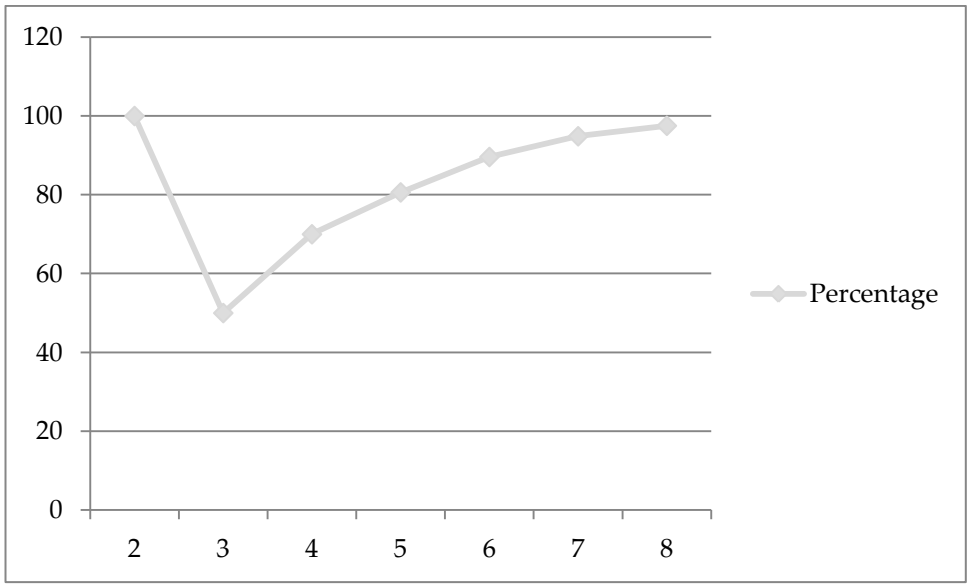


Figure 4.3: The percentage of solutions with m=0 for different n values

We implement four different versions of exponential search to see the limitations. These searches are isomorphic, non-isomorphic, feed-forward-free isomorphic and feed-forward-free non-isomorphic.

Figure 4.4 shows that over 93 percent of the total solutions are eliminated by our isomorphism elimination algorithm for n=8. This indicates that our isomorphism algorithm works reasonably well.

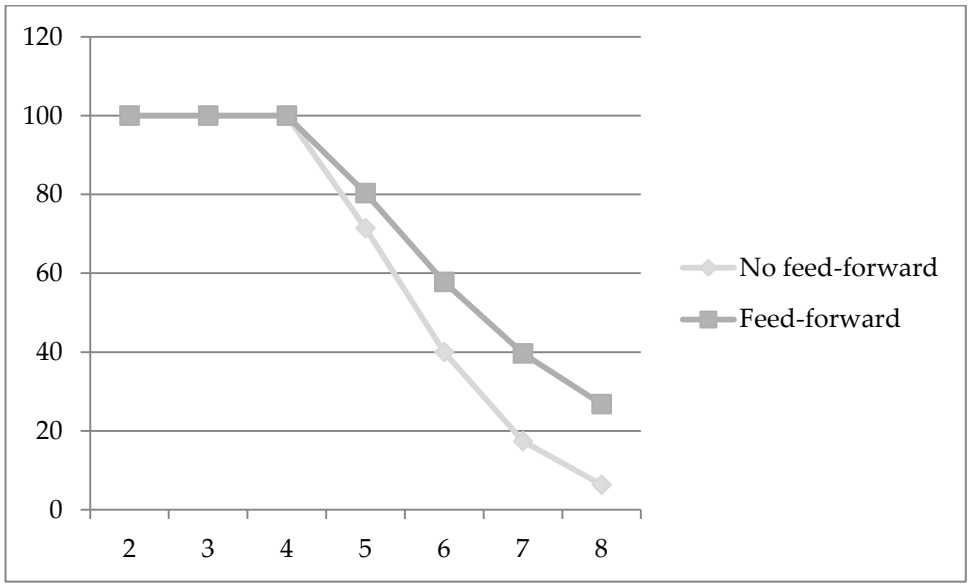


Figure 4.4: The percentage of non-isomorphic solutions for different *n* values.

Our isomorphism elimination method is designed for the limitations of this algorithm. For this reason, it works better when the solutions do not have feed-forward loops as seen in Figure 4.4.

4.2 Genetic Algorithm

We have used cross-over probability as 0.95 and mutation probability for each gene as 0.001. These are relatively high probabilities; because we need to find as many as possible results and the modification functions increase the diversity of the pool. We used a pool of population 100000 for the experiments unless another value is stated.

Most of the distinct results are found in the early generations. Especially for higher *m* values, few generations are enough to find most of the topologies. This shows that our pool generation heuristics work well. However our algorithm continues to find new results in later generations and reaches to a convergence state before finding all of the results. As seen in the Figure 4.5, some runs yield even fewer results for higher generations. This situation is called premature convergence. Changing the parameters of the modification functions or throwing new random elements to the pool may overcome this problem.

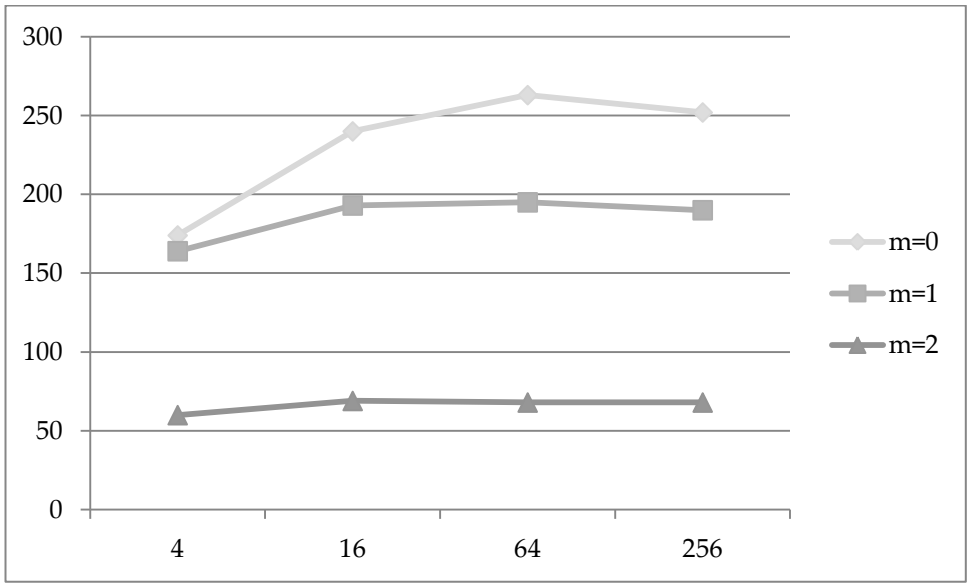


Figure 4.5: The number of solutions after several generations for *n*=8 values.

Changing the number of pool elements affect the number of solutions and premature convergence. In Figure 4.6, we can see that when the higher pool size gives us higher number of results as well as later convergence. In a normal situation, the number of results for pool size of 10^5 after 40 generations would be near to the number of results for pool size of 10^6 after 4 generations.

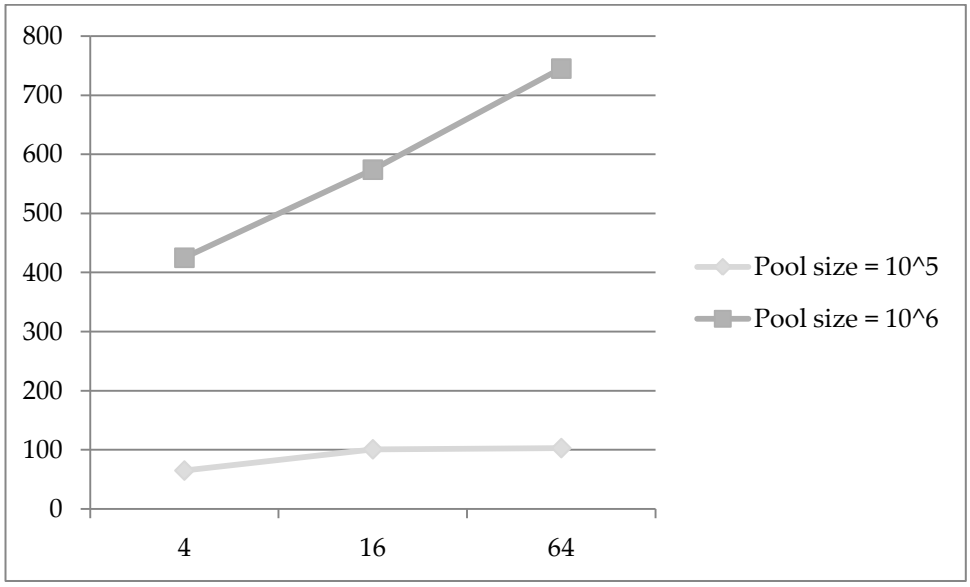


Figure 4.6: The number of solutions after several generations for n=12, m=0.

Our algorithm works in relatively fast time with respect to Kaderali *et al.*[3] which has the same network limitations. Their algorithm works in about 126 minutes with a more potent computer and a network of size 8.

CHAPTER 5

Conclusions and Future Work

5.1 Conclusions

We have adapted genetic algorithms to solve a very simplified version of signaling pathways. It is the first usage of genetic algorithms in signaling pathways as long as we have investigated. Our algorithm can be used as a base of a classification algorithm on definite topologies. Furthermore the results can be used to design new RNAi experiments to minimize the vast number of different topologies.

Although our algorithm works admissibly fast for a network construction problem, it has still problem of premature convergence in some cases. This problem can be overcome by tuning the parameters or using dynamic parameters in genetic algorithms.

Our graph isomorphism elimination algorithm is specially designed for the problem our problem. Although it can be used for all directed acyclic graphs the results show that our algorithm works better on our more limited graph schemes.

By using real valued edge weights instead of Boolean edges we approach to a less simplified problem. It also covers deactivation interactions. By slightly modifying our algorithm, we propose an approach to solve the real problem with most accurate result instead of finding lots of topologies that are consistent with data. We

do not have biological results for this problem. Therefore we cannot test both the model and the modified genetic algorithm. Also examining the methods that are used to combine neural networks with genetic algorithms will be beneficial [5].

5.1 Future Work

The ideas that can be followed but not examined enough in this thesis are:

1. A parallel algorithm can be designed to solve these problems because genetic algorithms are highly parallelizable.
2. The weighted problem and model should be tested using experimental wet-lab data.
3. A model is offered for counting the number of solutions. But it does not completely solve the whole counting problem. The remaining part is a challenging combinatorics problem.

REFERENCES

[1] Markowetz, F., Bloch, J., Spang, R (2005). Non-transcriptional pathway features reconstructed from secondary effects of RNA interference. *Bioinformatics vol 21; number 21,* 4026-4032

[2] Moffat J., Sabatini D. M. (2006). Building mammalian signalling pathways with RNAi screens. *Nature reviews molecular cell biology, vol 7; number 3*, 177-187

[3] Kaderali L., Dazert E., Zeuge U., Frese M., Bartenschlager R. (2009). Recontructing Signaling Pathways from RNAi Data using Probabilistic Boolean Threshold Networks. *Bioinformatics, 25(17)*, 2229-2235.

[4] Ourfali O., Shlomi T., Ideker T., Ruppin E. (2007). SPINE: A framework for signaling-regularoty pathway inference from cause-effect experiments. *Bioinformatics 23 (ISMB'07)*, I359-I366.

[5] Zhu D., Rabbat M., Hero A., Nowak R., Figueiredo M. (2006), De novo signaling pathway reconstruction from multiple datasources, *New Research in Signal Transduction.*

[6] Acharya, L., Judeh, T., Duan, Z., Rabbat, M., Zhu, D.. GSGS: A computational framework for reconstructing signaling pathways from gene sets. *Under review*.

[7] Markowetz F., Spang R. (2007). Inferring cellular networks - a review. *BMC Bioinformatics, 8(Suppl 6):S5*.

[8] Retrieved June 29, 1011 from http://en.wikipedia.org/wiki/Cell_signaling

[9] Retrieved June 29, 1011 from http://en.wikipedia.org/wiki/File:Signal_transduction_pathways.png

[10] Fire A., Xu S., Montgomery M. K., Kostas S. A., Driver S. E., Mello C. C. (1998). Potent and specific genetic interference by double-stranded RNA in Caenorhabditis elegans. *Nature, 391*, 806–811.

[11] Retrieved June 29, 1011 from http://en.wikipedia.org/wiki/File:RNAi-simplified.png

[12] Retrieved June 29, 1011 from http://en.wikipedia.org/wiki/Graph_isomorphism

[13] Zemlyachenko, V. N.; Korneenko, N. M.; Tyshkevich, R. I. (1985), Graph isomorphism problem, *Journal of Mathematical Sciences 29 (4)*.

[14] Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press. (Second edition: MIT Press, 1992.)

[15] Whitley, L. D. (1994). A genetic algorithm tutorial, *Statistical. Computing. vol. 4,* 65 - 85.

[16] Bäck, T., Hoffmeister, F., and Schwefel, H. (1991). A survey of evolution strategies. In R. K. Belew and L.B. Booker, eds., *Proceedings of the Fourth International Conference on Genetic Algorithms.* Morgan Kaufmann.

[17] Montana, D. J., and Davis L. D., (1989). Training feedforward networks using genetic algorithms. *In Proceedings of the International Joint Conference on Artificial Intelligence*. Morgan Kaufmann.

APPENDICES

Appendix A: Derivation of Equation 2

Derivation of Equation 2 from Equation 1 is as follows:

$$m = \sum_{k=2}^{n-1} \prod_{i=1}^{k-1} \prod_{j=k+1}^{n} 1 - p^{j-i}$$

In Equation 1 all the single probabilities i.e. p^{j-i} terms are taken as single variables. By using the combined nonexistence probability of each edge over every vertex expected value of m is calculated. First we assume that p<0.5 to make graph sparse. But the probability of existing more than one edge over a vertex is very low when p is lower than 0.5. We transform the equation with some negligence as:

$$n - 2 - m = \sum_{k=2}^{n-1} \sum_{i=1}^{k-1} \sum_{j=k+1}^{n} p^{j-i}$$

When p<0.5 this equation yields:

$$n - 2 - m = (\frac{p}{1-p})^2((n-2)(1-p^{n-1}) - 2p\frac{1-p^{n-2}}{1-p})$$

Since $(1-p^{n-1})$ and $(1-p^{n-2})$ are closed to 1, we ignored these two terms so the equation turn to:

$$n - 2 - m = r^2(n - 2 - 2r)$$

where r=p/(1-p). If we ignore the 2r term in the parenthesis we yield:

$$r = \sqrt{\frac{n-2-m}{n-2}}$$

At last we derive *p* from *r* and yield Equation 2 which is:

$$p = 1 - \frac{1}{1 + \sqrt{\frac{n-2-m}{n-2}}}$$

Appendix B: Test Results

In this appendix we present the results of the tests. Time values are shown as seconds. Columns denote *n* values and rows denote *m* values.

Table B.1: Exhaustive search results without isomorphism elimination.

	2	**3**	**4**	**5**	**6**	**7**	**8**
Time	0.01	0.01	0.01	0.01	0.02	0.27	36.73
Total	1	1	2	7	40	357	4824
0	1	0	1	4	28	283	4141
1		1	0	2	8	57	574
2			1	0	3	12	87
3				1	0	4	16
4					1	0	5
5						1	0
6							1

Table B.2: Exhaustive search results with isomorphism elimination.

	2	**3**	**4**	**5**	**6**	**7**	**8**
Time	0.01	0.01	0.01	0.01	0.02	0.27	38.43
Total	1	1	2	5	16	62	305
0	1	0	1	2	8	34	192
1		1	0	2	4	17	72
2			1	0	3	6	27
3				1	0	4	8
4					1	0	5
5						1	0
6							1

Table B.3: Exhaustive search results without isomorphism elimination and feedforward loops.

	2	**3**	**4**	**5**	**6**	**7**	**8**
Time	0.01	0.01	0.01	0.01	0.02	0.1	11.71
Total	1	2	10	122	3346	196082	23869210
0	1	1	7	103	3097	189241	23472463
1		1	2	15	220	6449	386118
2			1	3	24	352	10077
3				1	4	34	500
4					1	5	45
5						1	6
6							1

Table B.4: Exhaustive search results with isomorphism elimination and feedforward loops.

	2	**3**	**4**	**5**	**6**	**7**	**8**
Time	0.01	0.01	0.01	0.01	0.02	0.61	89.84
Total	1	2	10	98	1934	77750	6394488
0	1	1	7	79	1733	73757	6236105
1		1	2	15	172	3673	152086
2			1	3	24	280	5841
3				1	4	34	404
4					1	5	45
5						1	6
6							1

For the tests below, columns denote the number of generations.

Table B.5: Genetic algorithm results without isomorphism elimination.

n=8	**4** **4-Distinct** **4-Time**			**16** **16-Distinct** **16-Time**			**64** **64-Distinct** **64-Time**			**256** **256-Distinct** **256-Time**		
0	280	174	1.64	503	240	5.69	573	263	21.57	541	252	84.02
1	363	164	1.53	860	193	5.38	878	195	20.21	866	190	82.73
2	191	60	1.48	555	69	4.99	632	68	16.96	651	68	65.47
3	56	16	1.39									
4	20	5	1.36									

Table B.6: Genetic algorithm results with isomorphism elimination.

n=8	**4** **4-Distinct** **4-Time**			**16** **16-Distinct** **16-Time**			**64** **64-Distinct** **64-Time**		
0	434	62	1.66	699	73	5.67	751	70	21.89
1	927	46	1.52	1754	49	5.36	1763	47	20.4
2	1363	24	1.5	2885	24	4.88	2931	24	17.15
3	1825	6	1.46						
4	20	5	1.25						

Table B.7: Genetic algorithm results with isomorphism elimination for different pool sizes.

n=12, m=0	**4** **4-Distinct** **4-Time**			**16** **16-Distinct** **16-Time**			**64** **64-Distinct** **64-Time**		
Pool size = 10^5	71	65	2.44	107	101	8.49	109	103	32.94
Pool size = 10^6	502	425	25.46	760	574	90.45	1026	745	351.13

Table B.8: Genetic algorithm results without isomorphism elimination and feedforward loops

n=10	**5 generations**	**25 generations**	**125 generations**
m=0	31784	129867	240323
m=2	33191	92675	121285
m=4	3997	5918	6097
m=6	70	70	70