FBNNet Package Introduction

Leshi Chen

Contents

Introduction

FBNNet is an R package that provides mechanisms to draw insights into gene activation, inhibition, and protein decay. The package was implemented based on the novel Boolean model: fundamental Boolean model (FBM) to infer the genetic networks. This document introduces the R package and includes examples for use cases.

Use Cases

Case 1: The mammalian Cell Cycle

Many new algorithms can be verified using the simulated datasets derived from several known regulatory networks, and the results can be compared with other known regulatory networks. In this paper, we proposed to use the Example network, and the mammalian cell cycle network listed in Table 1, to generate test data to prove the concept of the proposed Fundamental Boolean Model and Networks. The mammalian cell cycle network demonstrated in [1]. To generate the experimental data, firstly, we used the command *loadNetwork* from the R package, BoolNet, to load the two networks specified in the two text files: example.txt and cellcycle.txt as shown in Table 1. Secondly, we used the command *generateTimeSeries* from the R package, BoolNet, to generate the time series data with the output from the previous step, with all default settings, i.e., the parameter *type* is synchronous, the parameter *noiseLevel* is 0, and the parameter perturbations is 0. The BoolNet package was demonstrated in the tutorial of [1] and the study of [2]. Figure 1 presents one of the generated time series data of the cell cycle network demonstrated in [1].

Table 1 Experimental Data

File Name: cellcycle.txt, Network Type: mammalian cell cycle network
targets, factors CycD, CycD Rb, (! CycA & ! CycB & ! CycD & ! CycE) | (p27 & ! CycB & ! CycD) E2F, (! Rb & ! CycA & ! CycB) | (p27 & ! Rb & ! CycB) CycE, (E2F & ! Rb) CycA, (E2F & ! Rb & ! Cdc20 & ! (Cdh1 & UbcH10)) | (CycA & ! Rb & ! Cdc20 & ! (Cdh1 & UbcH10)) p27, (! CycD & ! CycE & ! CycA & ! CycB) | (p27 & ! (CycE & CycA) & ! CycB &! CycD) Cdc20, CycB Cdh1,(! CycA & ! CycB) | (Cdc20) | (p27 & ! CycB)

UbcH10, ! Cdh1 | (Cdh1 & UbcH10 & (Cdc20 | CycA | CycB)) CycB, ! Cdc20 & ! Cdh1
File Name: example.txt, Network Type: Example Network
targets, factors Gene1, Gene1 Gene2, Gene1 & Gene5 & !Gene4 Gene3, Gene3 Gene4, Gene3 & !(Gene1 & Gene5) Gene5,!Gene2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
CycD	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Rb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E2F	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0
CycE	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0
CycA	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0
p27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Cdc20	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	1	1
Cdh1	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1
UbcH10	0	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	1	1
CycB	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0

Figure 1 One of the generated time series data for the Cell cycle network.

The R package *FBNNet* also contains a few pre-built data for the demonstration of the mammalian cell cycle network that has been described and discussed in the result section of the paper. The following contains the scripts to load the pre-built data:

```
loadExperimentData<-function()
{
  #to remove all variables we can use the following command
  # rm(list=ls(all=TRUE))
  print("load synchronous training series data -> synchronoustrainingseries. \n\n")
  load("synchronoustrainingseries.RDATA",envir = parent.frame())
  print("load synchronous cellcycle cube -> synchronous_cellcycle_cube. \n\n")
  load("synchronous_cellcycle_cube.RDATA",envir = parent.frame())
  print("load asynchronous training series data -> asynchronoustrainingseries. \n\n")
  load("asynchronoustrainingseries.RDATA",envir = parent.frame())
  print("load asynchronous cellcycle cube -> asynchronous_cellcycle_cube. \n\n")
  load("asynchronous_cellcycle_cube.RDATA",envir = parent.frame())
  print("load leukeamia data -> leukeamia. \n\n")
  load("leukeamia.RDATA",envir = parent.frame())
  print("load cellcycle genes -> cellcyclegenes \n\n")
  load("cellcyclegenes.RDATA")
  genes<<-rownames(synchronoustrainingseries[[1]])
  synfbnnetwork<<-mineFBNNetwork(synchronous_cellcycle_cube,genes)
  asynfbnnetwork<<-mineFBNNetwork(asynchronous_cellcycle_cube,genes)
  startstates<<-lapply(synchronoustrainingseries,function(x)x[,1])
  attractor<<-searchForAttractors(synfbnnetwork,startstates,genes)
}
```

Here are the scripts to construct the analysis cube, mine regulatory networks and visualize the generated fundamental Boolean Networks. To use the R package FBNNet, you need to download the binary version of FBNNet from https://github.com/clsdavid/FBNNet_Lincoln and install it.

To load the package, we use the command:

>require(FBNNet)

To demonstrate the example of Cell Cycle, we reuse some commands from the R package BoolNet, and hence, we load the BoolNet package as well:

>require(BoolNet)

Firstly, we use the command loadNetwork from the BoolNet package to load the traditional Boolean network from the text file: cellcycle.txt

> cellcyclenetwork<-loadNetwork("testthat/others/cellcycle.txt")

After that, we use the command generateTimeSeries command from the BoolNet package to generate test data (2^10 samples):

>trainingseries<-generateTimeSeries(cellcyclenetwork,10000,43)

The generated test data contains many duplicated samples, and hence, we use the command FBNDataReduction from the R package FBNNet to remove duplicate samples.

>trainingseries<-FBNDataReduction(trainingseries)

We also need initial states to find attractor, i.e., get the first state from all samples

> startstates<-lapply(trainingseries,function(x)x[,1])

Now, we start to construct the FBM cube for the mammalian cell cycle.

>cube<-
constructFBNCube(cellcyclenetwork$genes,cellcyclenetwork$genes,trainingseries,4,1,TRUE)

When the cube has built, we can mine the FBN type of network

> FBNcellcyclenetwork<-mineFBNNetwork(cube,cellcyclenetwork$genes)

The generated network can be plotted with the FBNNetwork Graph commands

>FBNNetwork.Graph(FBNcellcyclenetwork)

Figure 2 presents the result of the FBN of the mammalian cell cycle.

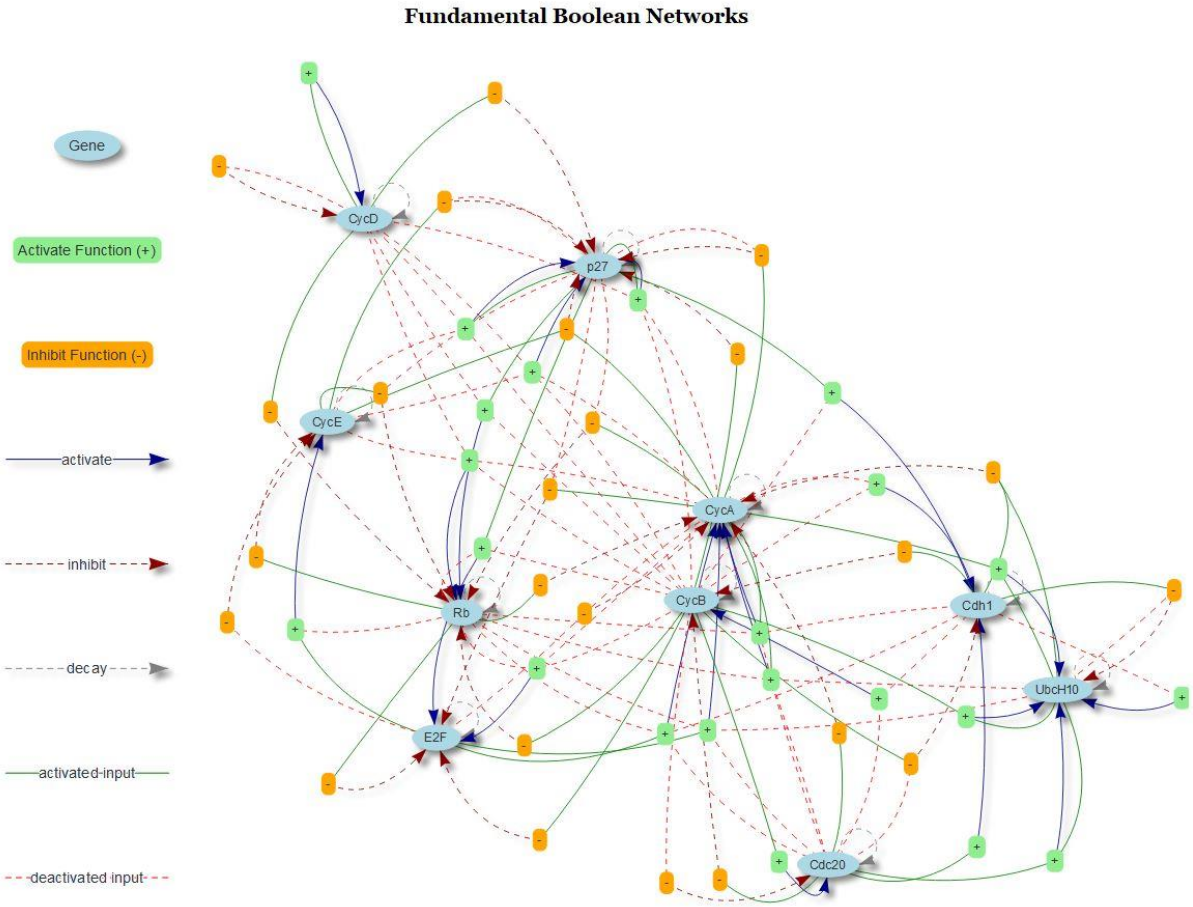

Figure 2 FBN of the mammalian cell cycle

We also can print out the Network

>print(FBNcellcyclenetwork)

Table 2 displays the novel Boolean network FBN of the mammalian cell cycle.

Table 2 FBN of the mammalian cell cycle under the FBM

```
Fundamental Boolean Network with  10 genes
Genes involved:
CycD, Rb, E2F, CycE, CycA, p27, Cdc20, Cdh1, UbcH10, CycB

Multiple Transition Functions for CycD with decay value = 1:
CycD_1_Activator: CycD = CycD (Confidence: 1, TimeStep: 1)
CycD_2_Inhibitor: CycD = !CycD (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for Rb with decay value = 1:
Rb_1_Activator: Rb = !CycD&p27&!CycB (Confidence: 1, TimeStep: 1)
Rb_2_Activator: Rb = !CycD&!CycE&!CycB&!CycA (Confidence: 1, TimeStep: 1)
Rb_3_Inhibitor: Rb = CycD (Confidence: 1, TimeStep: 1)
Rb_4_Inhibitor: Rb = CycB (Confidence: 1, TimeStep: 1)
Rb_5_Inhibitor: Rb = CycA&!p27 (Confidence: 1, TimeStep: 1)
Rb_6_Inhibitor: Rb = CycE&!p27 (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for E2F with decay value = 1:
E2F_1_Activator: E2F = !Rb&!CycA&!CycB (Confidence: 1, TimeStep: 1)
E2F_2_Activator: E2F = !Rb&p27&!CycB (Confidence: 1, TimeStep: 1)
E2F_3_Inhibitor: E2F = Rb (Confidence: 1, TimeStep: 1)
```

```
E2F_4_Inhibitor: E2F = CycB (Confidence: 1, TimeStep: 1)
E2F_5_Inhibitor: E2F = CycA&!p27 (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for CycE with decay value = 1:
CycE_1_Activator: CycE = !Rb&E2F (Confidence: 1, TimeStep: 1)
CycE_2_Inhibitor: CycE = !E2F (Confidence: 1, TimeStep: 1)
CycE_3_Inhibitor: CycE = Rb (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for CycA with decay value = 1:
CycA_1_Activator: CycA = !Rb&E2F&!Cdc20&!UbcH10 (Confidence: 1, TimeStep:
 1)
CycA_2_Activator: CycA = !Rb&CycA&!Cdc20&!UbcH10 (Confidence: 1, TimeSte
p: 1)
CycA_3_Activator: CycA = !Rb&CycA&!Cdc20&!Cdh1 (Confidence: 1, TimeStep:
1)
CycA_4_Activator: CycA = !Rb&E2F&!Cdc20&!Cdh1 (Confidence: 1, TimeStep:
1)
CycA_5_Inhibitor: CycA = Rb (Confidence: 1, TimeStep: 1)
CycA_6_Inhibitor: CycA = Cdc20 (Confidence: 1, TimeStep: 1)
CycA_7_Inhibitor: CycA = !E2F&!CycA (Confidence: 1, TimeStep: 1)
CycA_8_Inhibitor: CycA = Cdh1&UbcH10 (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for p27 with decay value = 1:
p27_1_Activator: p27 = !CycD&!CycE&!CycB&!CycA (Confidence: 1, TimeStep:
1)
p27_2_Activator: p27 = !CycD&!CycA&!CycB&p27 (Confidence: 1, TimeStep: 1)
p27_3_Activator: p27 = !CycD&!CycE&!CycB&p27 (Confidence: 1, TimeStep: 1)
p27_4_Inhibitor: p27 = CycD (Confidence: 1, TimeStep: 1)
p27_5_Inhibitor: p27 = CycB (Confidence: 1, TimeStep: 1)
p27_6_Inhibitor: p27 = CycA&!p27 (Confidence: 1, TimeStep: 1)
p27_7_Inhibitor: p27 = CycE&!p27 (Confidence: 1, TimeStep: 1)
p27_8_Inhibitor: p27 = CycE&CycA (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for Cdc20 with decay value = 1:
Cdc20_1_Activator: Cdc20 = CycB (Confidence: 1, TimeStep: 1)
Cdc20_2_Inhibitor: Cdc20 = !CycB (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for Cdh1 with decay value = 1:
Cdh1_1_Activator: Cdh1 = Cdc20 (Confidence: 1, TimeStep: 1)
Cdh1_2_Activator: Cdh1 = !CycA&!CycB (Confidence: 1, TimeStep: 1)
Cdh1_3_Activator: Cdh1 = p27&!CycB (Confidence: 1, TimeStep: 1)
Cdh1_4_Inhibitor: Cdh1 = !Cdc20&CycB (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for UbcH10 with decay value = 1:
UbcH10_1_Activator: UbcH10 = !Cdh1 (Confidence: 1, TimeStep: 1)
UbcH10_2_Activator: UbcH10 = Cdc20&UbcH10 (Confidence: 1, TimeStep: 1)
UbcH10_3_Activator: UbcH10 = UbcH10&CycB (Confidence: 1, TimeStep: 1)
UbcH10_4_Activator: UbcH10 = CycA&UbcH10 (Confidence: 1, TimeStep: 1)
UbcH10_5_Inhibitor: UbcH10 = Cdh1&!UbcH10 (Confidence: 1, TimeStep: 1)

Multiple Transition Functions for CycB with decay value = 1:
CycB_1_Activator: CycB = !Cdc20&!Cdh1 (Confidence: 1, TimeStep: 1)
CycB_2_Inhibitor: CycB = Cdh1 (Confidence: 1, TimeStep: 1)
CycB_3_Inhibitor: CycB = Cdc20 (Confidence: 1, TimeStep: 1)
```

To get attractors, we need to get the rownames from a sample.

>genes<-rownames(trainingseries [[1]])

The package FBNNet provides a command for search attractors under the synchronous scheme, and Figure 2 exhibits the result.

>attractor<-searchForAttractors(FBNcellcyclenetwork,startstates,genes)

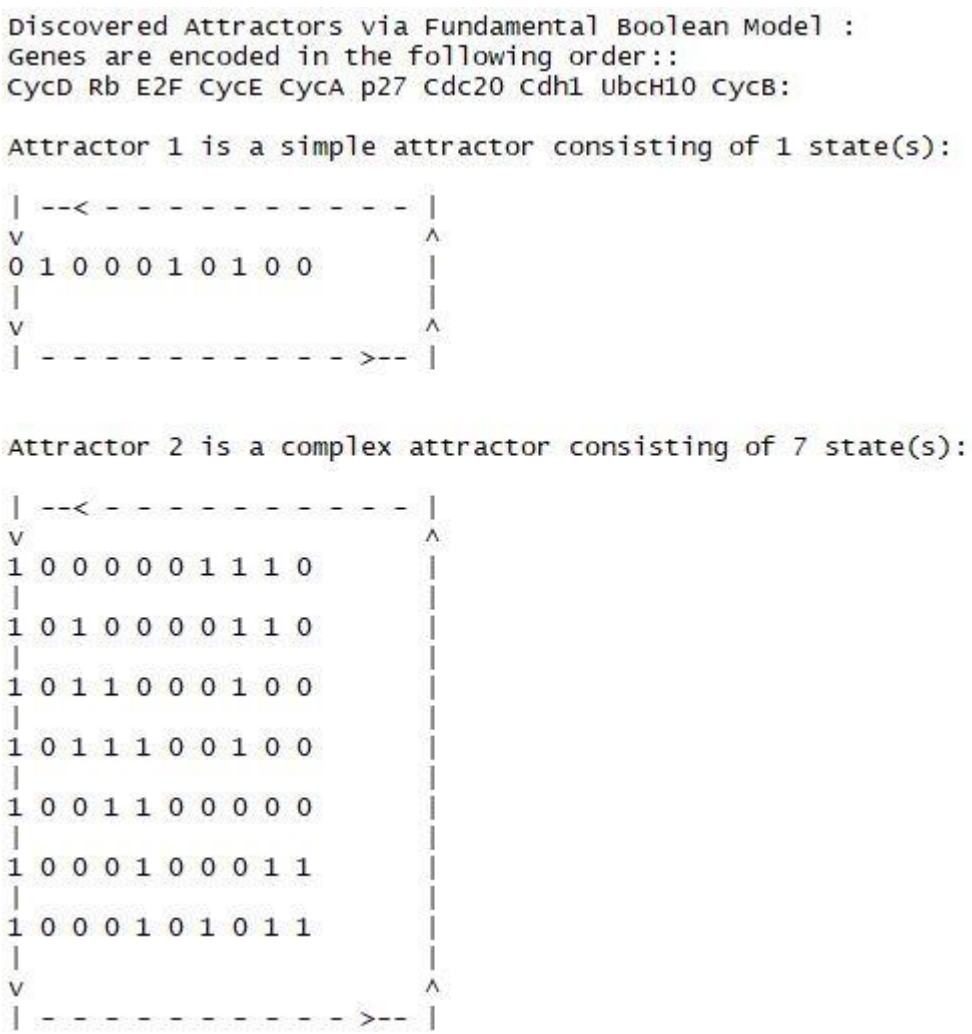

Figure 3 Attractors of the FBN of the mammalian cell cycle

The richness of the proposed model was its dynamic networks that provided a complete trajectory of gene activation, inhibition and protein decay. The package FBNNet also provides a command to plot the attractor in the way of dynamics. Figure 4 presents the result.

> FBNNetwork.Graph.DrawAttractor(FBNcellcyclenetwork,attractor,2)

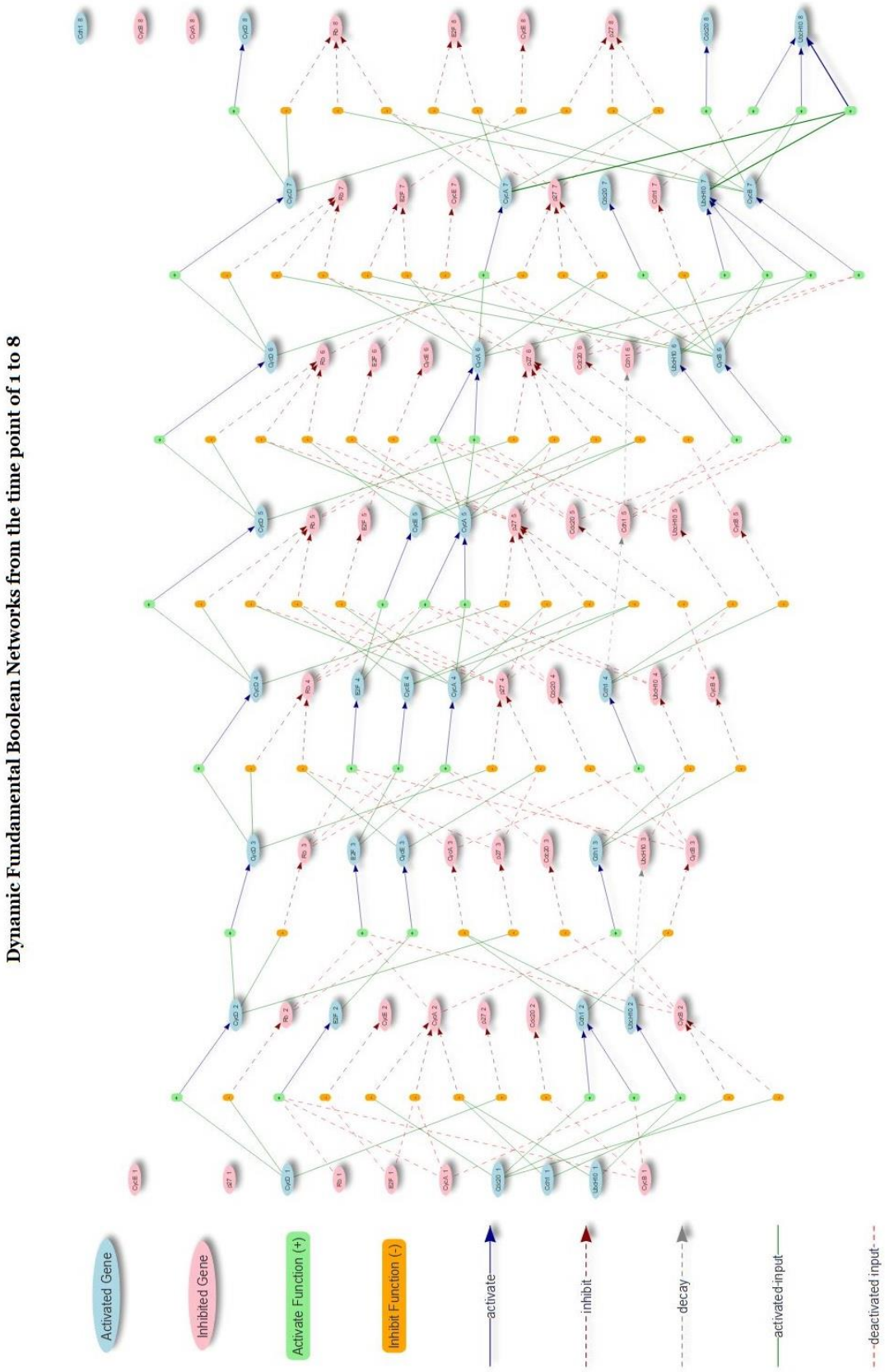

Figure 4 The dynamic trajectory of the attractor 2