CABERNET: a Cytoscape app for Augmented Boolean modEls of gene Regulatory NETworks
USER MANUAL

Andrea Paroni *[1], Alex Graudenzi[1], Giulio Caravagna[1], Giancarlo Mauri[1], and Marco Antoniotti[1]

[1]Dept. of Informatics, Systems and Communication, University of Milan-Bicocca

March 23, 2015

Contents

*to whom the correspondence should be addressed: *a.paroni@campus.unimib.it*

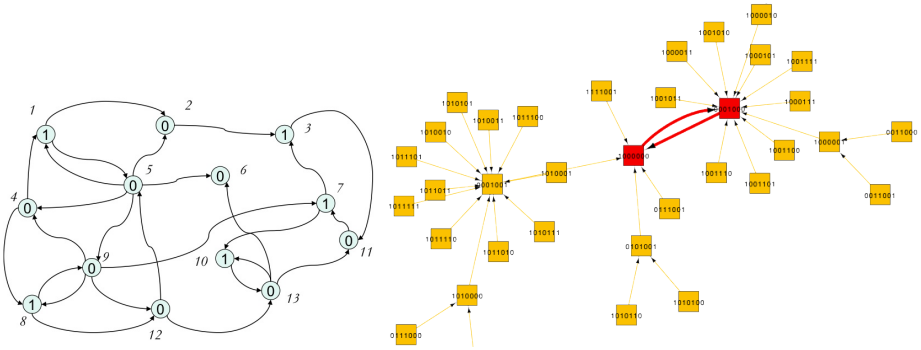

Figure 1: **Example RBN and attractors landscape.** In left, the RBN Boolean nodes represent genes (either active or inactive) and the edges regulatory pathways. A specific Boolean function (not shown here) is associated to each node. In right, each node represent a state of the system, i.e. the vector of the activation values of the genes, and the edges display the transitions between the states according to the deterministic dynamics.

1 Introduction: the theoretical framework

CABERNET is a Cytoscape 3.2.0 [17] app for the generation, the simulation and the analysis of dynamical models of *gene regulatory networks* (GRN). In this current version, GRNs are represented as *Noisy Random Boolean Networks* (NRBNs) [15, 16, 18].
Classical RBNs [13] are conceived to represent the complex processes of gene regulation in the simplest possible way: GRNs are represented as directed graphs in which Boolean nodes represent genes either synthesizing their specific protein/RNA or not, and regulating the activation of other genes in complex combinations . The value of the Boolean nodes is defined according to specific Boolean functions associated to each node, which depend on the value of input genes. In the original formulation the system is synchronous and deterministic and each node of the network is updated simultaneously at each discrete time step. Hence, after a certain transient the dynamical trajectory of the system will necessary end up in a cycle (at most of unitary length), which represent an attractor of the network (see Fig. 1). Different initial conditions lead to the same attractors, which represent different gene activation patterns that a cell with a unique genome is able to display, possibly standing for different phenotypic functions [12].

Since the model of RBNs is deterministic, random perturbations representing biological noise have been successively introduced in terms of temporary (i.e., flips) or permanent (i.e., knockin/knockout) perturbations, thus speaking of Noisy Random Boolean Networks. Perturbations allow to identify noise-induced transitions among the attractors, thus defining a stability matrix, also called *Attractor Transition Network* (ATN) (see Fig. 2). Following [18, 4] NRBNs allow to define a dynamical model of cell differentiation, i.e., the process according to which the progeny of stem cells becomes progressively more specialized by developing in different cell types. The model is general and is not referred to any specific organism or cell types and is grounded on the key concept of *emergent dynamical behaviour*. In particular, the differentiation properties are correlated with the dynamical properties of the underlying GRNs and, in particular, with the stability of their attractors in presence of biological noise. The general idea is that more differentiated cells would wander in a smaller portion of the phase space, because of more efficient control mechanism against possible genomic perturbations and random fluctuations [7, 11, 5, 6, 3, 10].

Formally, consider a NRBN with m attractors and let A_i (with $i \in [1, m]$) be the i-th

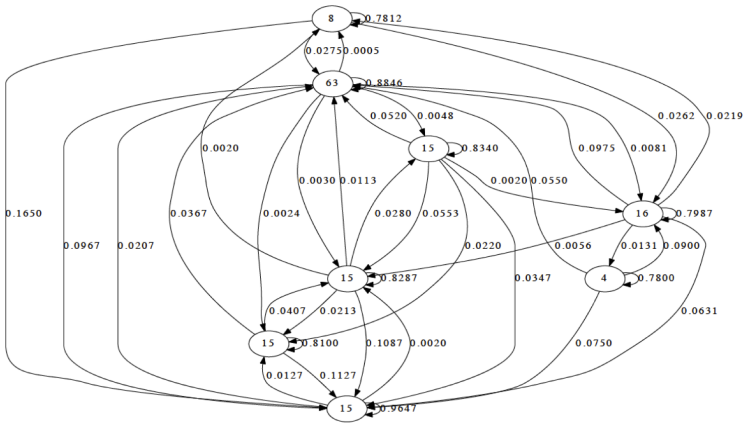

Figure 2: **Example Attractor Transition Network.** Each node represents a specific NRBN attractor, inscribed number is the attractor's length. The edges depict the transitions between attractors that occur after single-flip perturbations of nodes, with the corresponding frequency of occurrence.

attractor of the network and let A be the set of such attractors. An attractor A_j is directly δ-reachable from another A_i $(i, j \in [1, m], i \neq j)$ if at least a fraction δ of different flip perturbations on singles states of A_i leads the system, when it is in attractor A_i, to A_j. A Threshold ergodic set (TES) is a set of attractors with the following properties: 1) any attractor member of the TES is δ-reachable from any other attractor belonging to the set (it is an *ergodic set* in the ATN); 2) given that threshold value, there are no outgoing connections from any member of the TES toward an attractor not belonging to it. In the model TESs represent cell types, characterized by a specific differentiation degree as that indicated by the threshold, i.e., gene activation patterns in which the cell can wander due to random perturbations. So, at $\delta = 0$ one typically has a unique TES composed of many interconnected attractors, which represent less differentiated cells (e.g., toti- and multipotent stem cells) and which, by increasing the threshold, breaks into some disjoint and smaller TESs (i.e., intermediate stages), until, at the highest levels of the threshold, all the attractors are also TESs, standing for fully differentiated cells. In this way, it is possible to define a hierarchy among cell types, which identify a specific differentiation tree (see Figure. 3).

The model can reproduce, among other key properties: $[(i)]$ different *degree of differentiation*, i.e. from totipotent/multipotent stem cells to transit amplifying stages up to to fully differentiated cells [1]; $[(ii)]$ the phenomenon of *stochastic differentiation*, according to which a population of multipotent cells can generate progenies of different types, through a stochastic differentiation process [9, 2, 8]. For a more detailed description of the RBN and NRBN models please refer to the wide relative literature, as, e.g.,[14].

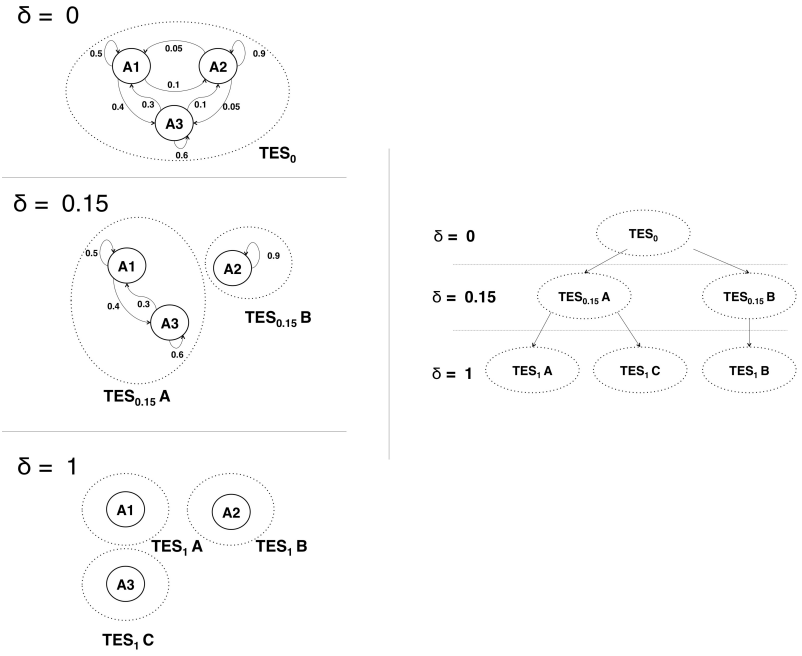

Figure 3: **Threshold-dependent ATN and the tree-like TES landscape.** The circle nodes are attractors of an example NRBN, the edges represent the relative frequency of transitions from one attractor to another one, after a 1 time step-flip of a random node in a random state of the attractor (performed an elevated number of times). In this case we show three different values of threshold, i.e.: $\delta = 0$, $\delta = 0.15$ and $\delta = 1$. TESs, i.e. strongly connected components in the threshold-dependent ATN are represented through dotted lines and the relative threshold is indicated in the subscripted index. In the right diagram it is shown the tree-like representation of the TES landscape, which determines the differentiation tree for this NRBN.

On the basis of this theoretical framework CABERNET presents a range of different simulation and analysis functions.

- **Network generation**.

  (*a*) Random networks with specified structural properties. Given the key parameters, CABERNET can generate and simulate ensembles of random network with shared features.

  (*b*) Augmentation of partial networks. Given partially defined networks as input, e.g., from publicly available dataset or known GRNs, CABERNET can augment the networks according to key structural and topological parameters.

- **Network simulation**. The attractor landscape and the properties of the attractors of the networks can be extensively investigated.

- **Network selection**. Given an input differentiation tree, the app allows to search for NRBNs whose emergent behaviour is in accordance with the input tree (in terms of the expected stability and dynamical trajectory). The app is based on a *generative approach*, i.e., NRBNs are randomly created according to user-defined features such as statistical properties and topologies, and a batch process accepts/discard the GRNs matching the input lineage commitment tree.

- **Robustness analysis**. Different kind of perturbations can be performed on the networks and the relative stability can be subsequently assessed via robustness analyses, for instance by analyzing measures such as *avalanches* and *sensitivity*.
- **Network analyses**. A wide range of statical and dynamical properties of the simulated network can be analyzed with CABERNET.
- **Visualization**. The powerful visualization capabilities of Cytoscape can be used to analyze the topological and dynamical properties of the simulated networks.

2 CABERNET installation and settings

CABERNET is fully tested with *Cytoscape version 3.2.0*. We do not provide support for other versions of the application. CABERNET is distributed under the terms of a BSD-like license, included in file `COPYING` of the software package. More information on CABERNET can be found at the project website:

`http://bimib.disco.unimib.it/index.php/CABERNET`.

CABERNET version 1.0 is released as a JAR archive

`CABERNET-1.0.jar`.

Two downloads options are possible: (*i*) download from CABERNET website and (*ii*) download from the Cytoscape App Store at

`http://apps.cytoscape.org/`.

Installation. CABERNET can be installed using the *App Manager* service reachable at

`Apps` → `App Manager` → `Install from file...`

Running a CABERNET session. CABERNET sessions are batch computations which depend on user-defined parameters. Parameters are given by a step-by-step Wizard procedure; input parameters include, for instance, a *differentiation tree*, the *NRBNs structure* and the *analysis settings*. In what follows we explain all the possible parameters for a CABERNET session.

3 CABERNET Wizard

The wizard is divided in six different sections, representing the main simulation and analysis stages, namely:

- Network Generation Mode
- Topology and Functions
- Sampling
- Experimental Settings
- Differentiation Tree Comparison
- Outputs

.

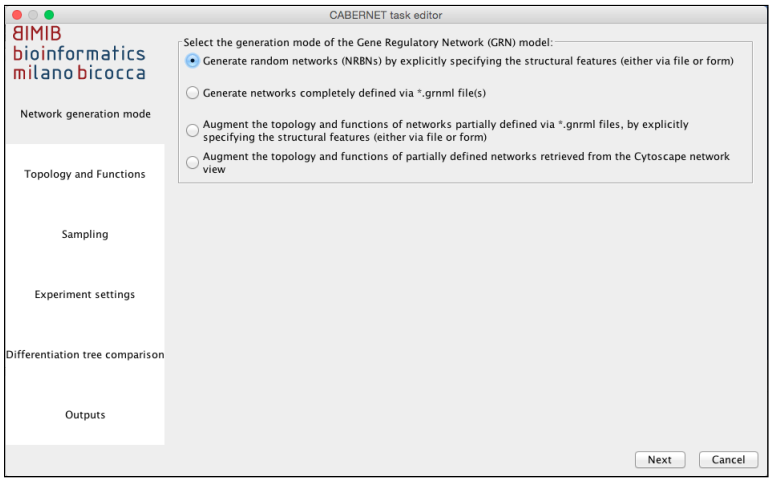

Figure 4: **Wizard A: Network Generation Mode**

3.1 Wizard A: Network Generation Mode

The first section of the Wizard, named Network Generation Mode, consists of different panels, as shown in Figure 4.

-PANEL 1-

In the first panel of the section the user is asked to define the network generation mode, among the following.

- **Generate random networks (NRBNs) by explicitly specifying the structural features (either via file or form)**. This option allows to specify all the network structural parameters for network generation. Each simulated network will be randomly created before the successive tasks.
- **Generate networks completely defined via *.grnml file(s)**. This option allows to load completely defined networks from GRNML files[1] that will be successively simulated. In this case, the user can specify only the experiment and output features. The user can select more than one network and the entire simulation process will be performed for each net.
- **Augment the topology and functions of networks partially defined via *.gnrml files, by explicitly specifying the structural features (either via file or form).** This option allows to load a partially defined network, which CABERNET will augment according to the specified structural features. We remark that an incomplete network is a network with only a subset of nodes, edges or functions defined. The user can select more than one network and the entire simulation process will be performed for each of them.
- **Augment the topology and functions of partially defined networks retrieved from the Cytoscape network view.** This option allows to load the network directly form the current selected view on Cytoscape. As in the previous case, the loaded network will be completed with other nodes or edges and functions as defined by the user in the succeeding steps.

[1] GRNML is the CABERNET and GRNSim (i.e., the stand alone terminal version of CABERNET) output format. This format stores all the network structure features, such as the network nodes, edges and functions

GRNML network definition file format An example of the file format for the specification of the network complete (or partial) structure and functions is the following:

```
<graph topology = "ScaleFree" nodes_number = "10">
<node id = "0" name = "gene_0" >
<function type = "canalizing" input_number = "2">
<input_node >5</input_node>
<input_node >8</input_node>
<input_node >3</input_node>
<canalizing_input> 5 </canalizing_input>
<canalizing_input> 8 </canalizing_input>
<bias> 0.5 </bias>
<entry input = "11" output = "1"></entry>
<entry input = "00" output = "0"></entry>
<entry input = "01" output = "1"></entry>
<entry input = "10" output = "1"></entry>
</function>
</node>

Other node definitions...

<edge source = "0" destination = "6"> </edge>
<edge source = "0" destination = "7"> </edge>
<edge source = "1" destination = "4"> </edge>

Other edge definitions...

</graph>
```

-PANEL 2-

When required, the user will be asked to insert the structural parameters to generate the networks, in two possible ways:

- (*From input form* option). In this case the user will be asked to insert the structural properties of the networks step-by-step with an interactive form.
- (*From *.txt text file* option). The features file is a "blank-separated" text file: in each row a property with its relative value are defined.

txt network features file format An example of the file format for the specification of the structural features of the networks is the following:

```
<simulation features file>
topology ScaleFree
nodes 100
algorithm BarabasiAlbert
k 3
ni 4
in-out-probability 0.5
completely-defined-functions no
```

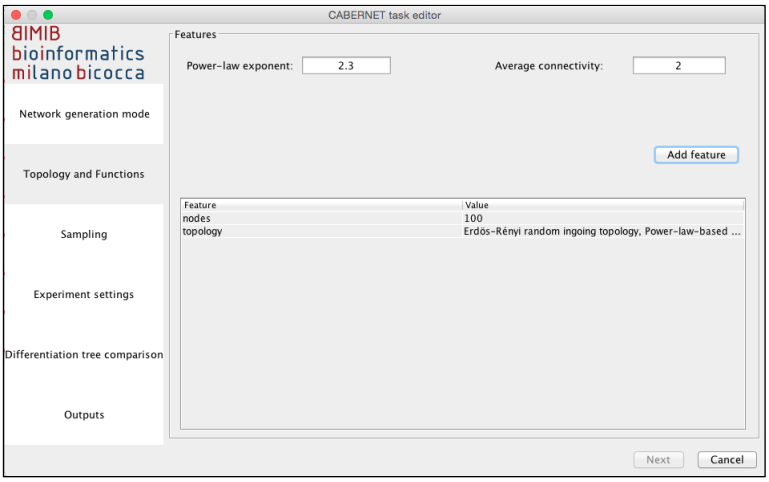

Figure 5: **Wizard B: Topology and Functions** panel 2

```
function-type Boolean
bias-type 0.4
bias-value 0.6
or-type 0.2
and-type 0.2
canalized-type 0.2
sampling-method Partial
initial-conditions 10000
max-simulation-times -1
how-many-nodes-to-perturb 1
ratio-of-states-to-perturb 0.9
mutation-type Flip
min-duration-of-perturb 1
max-duration-of-perturb 1
max-net-to-test 10000
atm-computation yes
tree-matching no
avalanches-sensitivity-computation no
unmatching-store no
```

-PANEL 3-

The number of networks to generate and simulate with common structural features will be included at this step.

3.2 Wizard B: Topology and Functions

If required, in the following section the topological properties of the networks and those relative to the updating functions will be defined [Figure 5]. In particular the input form includes the following parameters:

-PANEL 1-

Number of nodes. This property is used in order to define the total number of nodes of the network. If the network is partially defined, this number includes both the added and the original nodes.

-PANEL 2-

Network topology. This property is used in order to define the topology of the network. The topology defines the arrangement of the various elements of a network, such as nodes and edges.
CABERNET allows to use all the following topologies:

- *Erdös- Rényi random ingoing topology, Erdös- Rényi random outgoing topology*. If this topology has been selected, the succeeding required parameter is the *number of edges* to add. The implemented model uses directly the number of edges and not the edge probability.
- *Fixed number of inputs, Erdös- Rényi random outgoing topology*. If this topology has been selected, the succeeding required parameter are the *number of inputs* for every node. The actual number of added edges will be nodes*inputs.
- *Barabasi-Albert's preferential attachment (Scale-free)*. If this topology has been selected, the succeeding required parameters are the *number of initial nodes* (it must be between 0 and the number of nodes of the network), *the average connectivity* and the *ratio of incoming/outcoming edges*.
- *Erdös- Rényi random ingoing topology, Power-law-based outgoing topology (Scale-free)*. If this topology has been selected, the succeeding required parameters are the *Power Law exponent* and the *average connectivity*. Note that in this case the scale free distribution will be obtained as outcoming degree distribution.
- *Fixed number of inputs, Power-law-based outgoing topology (Scale-free)*. If this topology has been selected, the succeeding required parameters are the *average connectivity*, the *Power Law exponent* and the *number of inputs required* for all the nodes.
- *Watts-Strogatz small-world topology*. If this topology has been selected, the succeeding parameters are the *average connectivity* and the *edge switching probability* (this second parameter must be between 0 and 1).

-PANEL 3-

Updating functions. Once the topology of the network has been selected, the next step is the definition of the updating functions. First of all, it is necessary to select the type of the functions: in this version only boolean function are allowed.
Completely defined functions property means that each function will be completely defined (for all the possible input sequences the output value is produced) during the network creation process. Note that it is dispirited to set this property on for networks with high incoming degree nodes. Five types of functions are available:

- *Bias-based random functions*
- *And*
- *Or*

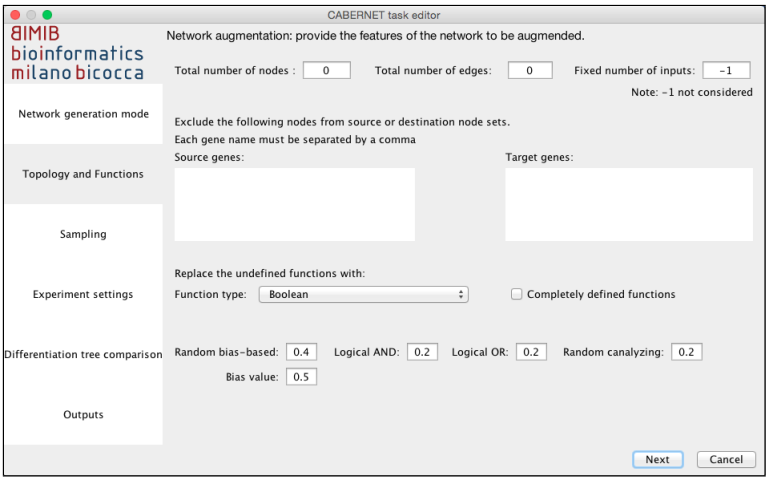

Figure 6: **Wizard B: Topology and Functions** for the network augmentation process

- *Canalizing functions*

In the wizard it is necessary to set the ratio of each function type to use. Note that the sum of all the ratios **must be equal to 1**. Note, for the augmentation process the *Topology and Functions* panel is lightly different from what presented above but the same information are required [Figure 6].

3.3 Wizard C: Sampling

Once the structure of the networks has been defined, the next step is the sampling settings [figure 7]. First of all, it is necessary to set all the parameters for the attractor search. There are two distinct strategies for the attractor search: (*a*) the *Brute force* and (*b*) the *Partial sampling*.

With the *Brute force* sampling method all the possible initial states of the system are simulated; on the other hand, with the *Partial sampling* method only a random subset of the possible initial states are tested. The first choice is advised only for small networks (note that in a Boolean network all the possible initial states are 2^{nodes}). For both the

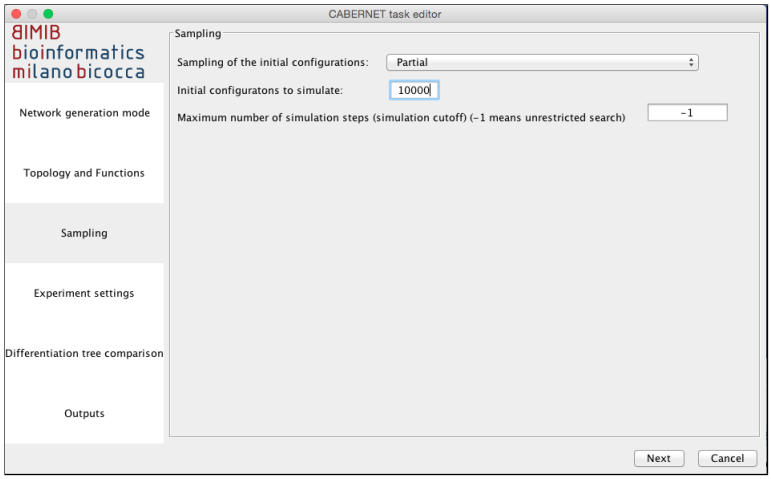

Figure 7: **Wizard C: Sampling**

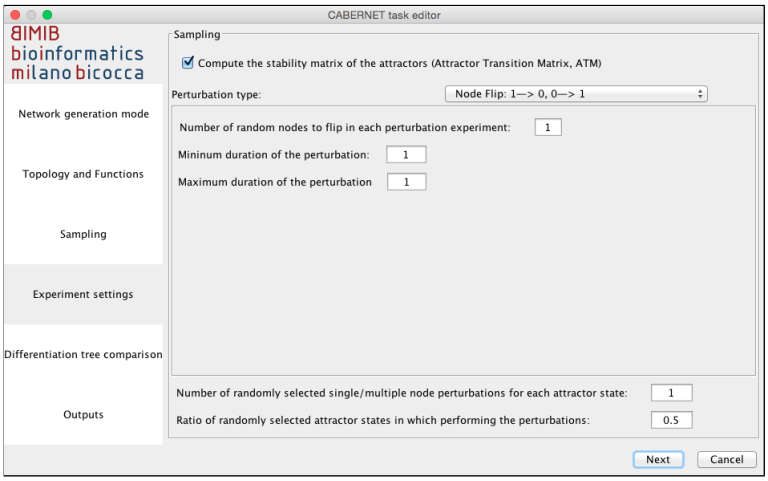

Figure 8: **Wizard D: Experiment settings** panel 1

strategies a cutoff value (number of iterations for a single initial state) is required, but it can be ignored if set equal to -1.
If the *Partial sampling* method is selected, it is necessary to set the number of initial states to test (it must be between 0 and 2^{nodes}).

3.4 Wizard D: Experiment settings

The next input frame is needed to define the perturbation parameters and the types of experiments to perform [Figure 8].

-PANEL 1-

First, the user is asked to choose whether compute the Attractor Transition Matrix (ATM), which represent the stability matrix describing the transitions among the attractors as a consequence of perturbations of various types.

- Type of perturbations. Two types of perturbations are possible:

  Flips: a flip is the negation of the value of a node (e.g., if the value of the is 0, the flip sets it to 1)

  Knock-in and *Knock-out*: regardless of the real value of the perturbed node, this perturbation sets it to 1 (i.e., knock-in) or to 0 (i.e., knock-out).

- The *ratio of attractors states to perturb* value expresses the number (in ratio) of states of each attractor to use for the perturbations experiments: this value must be between 0 and 1.

- The *Number of experiments for the same node* value expresses the number of times to repeat a perturbation experiment for the same state; this value must be greater than 0. If the selected perturbations type is flip, it is necessary to set the number of nodes to flip at the same time in the same experiment. It is also necessary to set the duration of the perturbations: for each experiment the duration is determined according to a uniform probability distribution between the minimum and maximum values. If the selected perturbations type is knock-in/knock-out, it is necessary to set separately the number of knocked-in and knocked-out nodes at the same time and their relative durations.

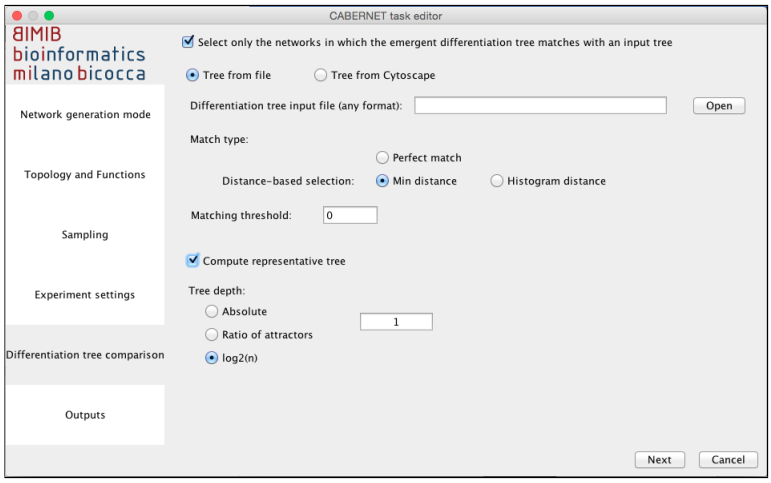

Figure 9: **Wizard E: Differentiation tree comparison**

3.5 Wizard E: Differentiation tree comparison

The tree matching input form is used in order to define all the parameters for the comparison of an input differentiation tree with the emerging tree, as defined on the basis of the stability of the attractors of the networks [figure 9].

This task is not mandatory and it is possible to select two different input methods for the tree: either (*a*) from file or from (*b*) the selected Cytoscape view.

Once the tree has been defined, it is important to choose the comparison distance to use:

- *Perfect matching*: the original and the built trees must be identical and in the worst case all the possible trees from the same network are tested. This method is extremely hard to perform. (NP-Complete problem [19]).
- *Threshold distance*: given a certain threshold and a distance type, between: (*a*) edit distance and (*b*) histogram distance, only those networks in which the distance between the emerging and the input tree is below the threshold are selected. Note: if the threshold value has been set to 0, this is equivalent to the perfect match.

In this section is also possible to select the *compute representative tree* features. If the function is selected, the representative tree of each network will be computed. For a more detailed description of this functionality see section 4.6.

Tree file format The file format for the tree input is a simple blank separated values file defined as follows:

```
<Differentiation tree description file>
level<blank>node_id<blank>parent
```

Example:

```
<Differentiation tree description file>
0 0 0
1 1 0
1 2 0
2 3 1
2 4 1
```

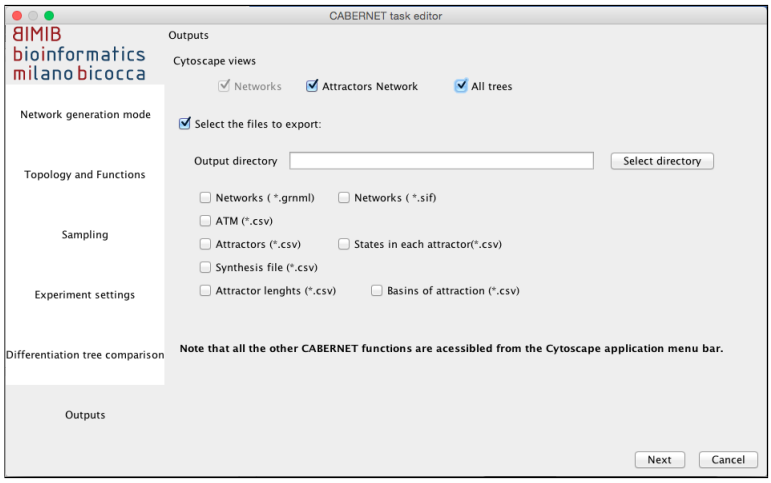

Figure 10: **Wizard F: Outputs**

3.6 Wizard F: Outputs

The last input form concerns the outputs to export [figure 10]. In particular it is possible to export separate files for the following objects:

- The network in a GRNML file. GRNML file is the standard input/output format for CABERNET.
- The network in a SIF file.
- The ATM matrix in a CSV file
- The attractors in a CSV file: each row is an attractor and the comma-separated sequences in each row are the attractor's states.
- States in each attractor in a CSV file: in each row there are the states, its attractor and its position in the basin of attraction. The first row of this file can be Partial or BruteForce and defines the sampling type used in the simulation.
- The synthesis file. It is a CSV file with some data and statistics: network id, clustering coefficient, average bias, average path length, network diameter, number of attractors in the network, average attractors lengths, the tree distance and the thresholds sequence (if the tree comparison is performed).
- The attractors lengths in a CSV file. Each row contains all the lengths of the found attractors.
- The basins of attraction sizes in a CSV file. Each row contains all the sizes of the basins of attraction.

The exporting frame can be reached from the right-click network menu selecting `Apps/Export (CABERNET)` command or from `Apps/CABERNET/Export option`.
It is possible to export only the selected network or all the networks at the same time.

It is also possible to visualise in the Cytoscape active window:

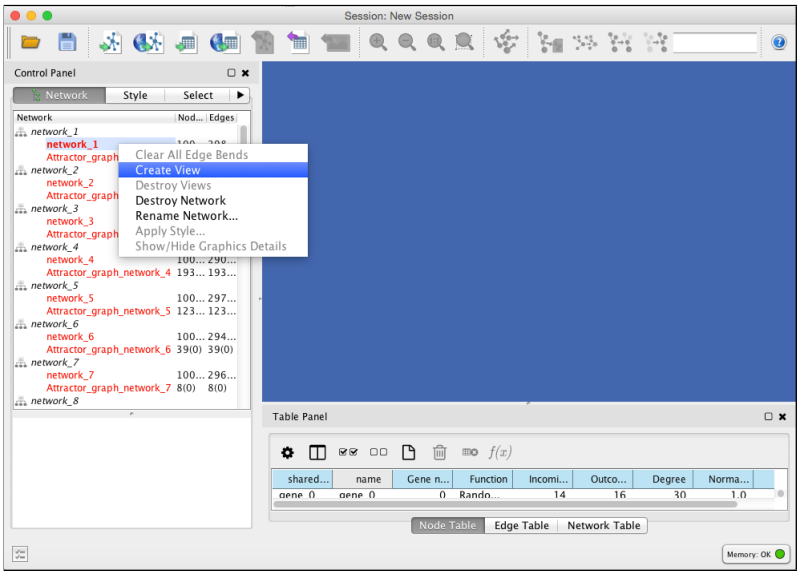

Figure 11: **Cytoscape Network Panel:** network view creation

- *Networks (Cytoscape views)* Creates a Cytoscape network object for every matching network.
- *Attractors network (Cytoscape views)* Creates a Cytoscape network object for every attractors network. An attractors network is a graph where the nodes are (N)RBN states and the edges connected two reachable states. This network shows the attractors and their states.
- *All trees (Cytoscape views)* Creates a view for all the representative tree that can be obtained from each network. This feature is visible only if the *compute representative tree* property is set in the *Wizard E: Differentiation tree comparison* panel [see section 3.5].

4 CABERNET functions in Cytoscape Active Window

Once the networks have been created and the simulation is completed, several important functions implemented in CABERNET are available directly in the Cytoscape active window.

4.1 Cytoscape Network Panel

In the network panel all the generated networks are listed, namely:

1. the simulated NRBNs
2. the relative Attractor networks, i.e., the representation of all the network configurations belonging to the attractors that have been found during the simulation. Each node in this network is a network configuration (i.e., a state vector) and each edge represents a transition between states.

In order to create the Cytoscape view it is necessary to right-click the selected network and choose the `Create View` command [Figure 11].

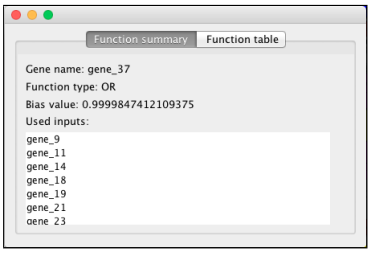

Figure 12: **Functions Exploration**

4.2 Functions Exploration

This function allows the user to analyze the details of a specific updating function. by selecting a node of the chosen network, and in the right-click menu selecting `Apps/Explore node function (CABERNET)`. The same function is accessible from `Apps/CABERNET/Functions/Explore node function` function.
The explores function view is composed by two tabs: in the first, *Function summary*, some general information about the selected function, likes the type and the inputs, is provided. In the second tab, *Function table*, there is the explicit representation of the function [Figure 12].
Note that the order of the genes in the input sequence represented in the truth table is the same defined in the summary tab.

4.3 ATM view

This function allows the user to display the Attractor Transition Matrix (ATM) of the selected network. The ATM describes all the possible transitions among attractors that are induced by (user-defined) perturbations. In order to show the ATM view it is necessary to choose a network view and select the `Apps/Show ATM (CABERNET)` right-click menu command. The same functionality can be accessed from `Apps/CABERNET/Functions/Show attractor stability analysis/Show Attractor Transition Matrix`.
In the ATM view, the ATM_0 shows the network with all the transition probabilities. This view offers also the possibility to create a Threshold Ergodic Set graph (TES graph) and a corresponding view with a specific threshold value: given a certain threshold, all the transitions that are characterized by a frequency below the threshold are removed and a pruned transition network is shown. Note that the threshold value must be between 0 and 1.
There are two different types of TES graphs: (*a*) the complete and (*b*) the collapsed one. In the complete graph all the states of each attractor are displayed as nodes and there are two types of edges: the state-to-state transition and the attractor-to-attractor transition. In the latter type, the edges show the transition probabilities and the edge width is proportional to that probability. The collapsed view shows each attractor as a unique node and only the transitions edges are displayed [Figure 13].

4.4 Styles

CABERNET allows to use several ad-hoc network styles selectable from the style tab in the control panel [Figure 14].

- CABERNET Network: this style has been thought for the representation of NRBNs. The color of the nodes is defined on the basis of the functions type and the size of

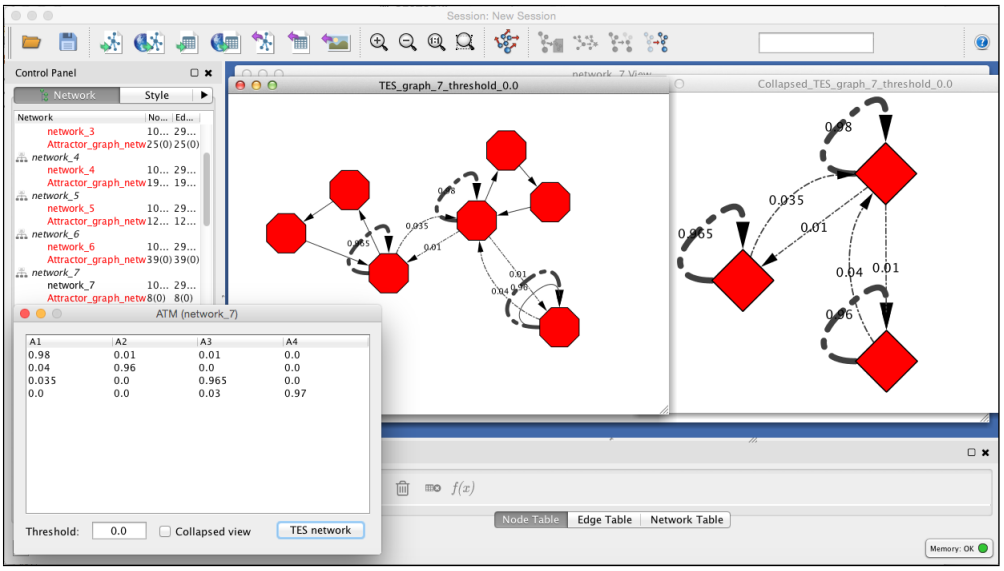

Figure 13: **ATM view:** view and TES graphs (not collapsed on the left and collapsed on the right)

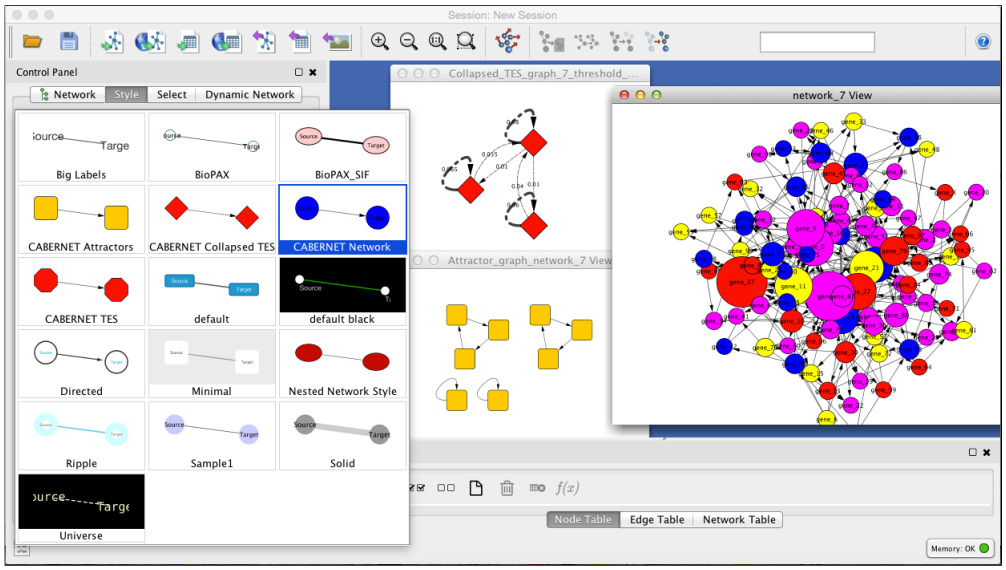

Figure 14: **Styles**

each node is proportional to its degree. The edges are oriented.

- CABERNET Attractors: this style has been thought for the attractors graph. The nodes are the network states that belong to a specific attractor and the edges correspond to transitions among states.
- CABERNET TES complete graph.
- CABERNET TES collapsed graph.

4.5 TES number chart

This function allows the user to visualize the variation of the number of TESs with respect to different values of threshold. It can be accessed from the right-click network menu selecting `Apps/Show threshold-dependent TES variation chart` command or from `Apps/CABERNET/Functions/Show attractor stability analysis/Show threshold-dependent TES variation chart`. It is possible to choose the threshold step and update the chart pressing the *Update chart* button [Figure 15].

4.6 Representative tree

This functionality allows the user to visualize the representative tree with a defined depth derivable from the selected network [figure 16].
The representative tree of a network is the most frequently tree with the required depth obtained from the network. The algorithm computes all the TES trees selecting all the possible ordered combinations of thresholds and, at the end, returns the most frequently emergent tree.
There are three different ways to define the required depth for the tree:

- **Absolute**: in this case, the inserted value is the real depth of the representative tree (the value must be between 1 and the number of the attractors).
- **Ratio of attractors**: in this case, the inserted value is a ratio of the number of the attractors discovered for the selected network (the value must be between 0 and 1). The real depth of the representative tree depends on the number of the attractors of the analyzed network.
- **log2(n)**: in this case, the representative tree depth is obtained as $\lg_2(n)$, where n is the number of attractors discovered in the selected network.

This functionality can be accessed from the right-click network menu selecting `Apps/Compute representative tree` or from `Apps/CABERNET/Functions/Compute representative tree`.

4.7 Dynamical properties analysis

This functionality allows the user to visualize several statistics regarding the dynamical behaviour of the networks: (*a*) the attractor length distribution, (*b*) the distribution

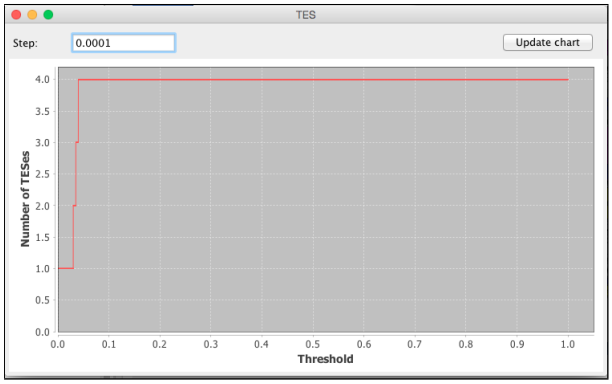

Figure 15: **TES number chart**

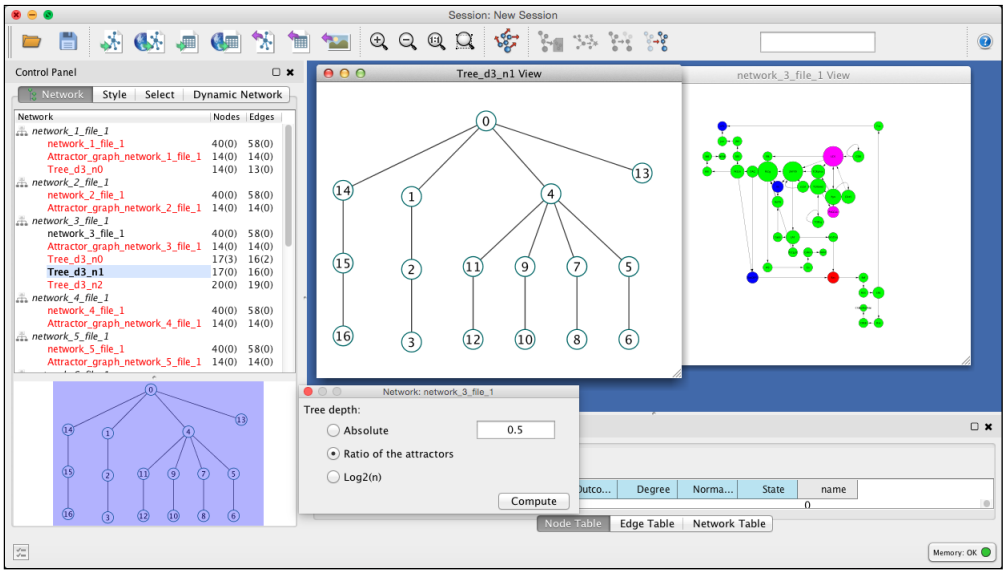

Figure 16: **Representative tree computation**

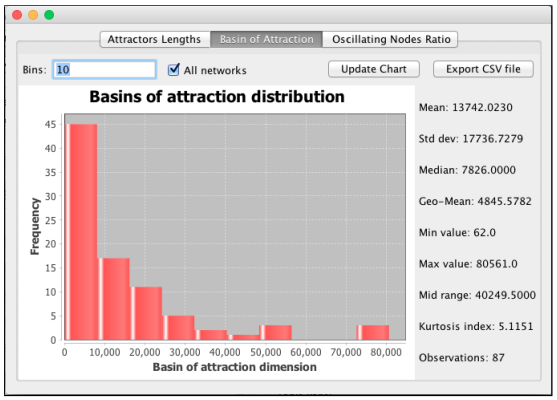

Figure 17: **Dynamical properties analysis**

of the basins of attraction and (c) the ratio of oscillating nodes over fixed-value nodes. This function can be accessed from the right-click network menu selecting `Apps/Show network dynamical properties` or from `Apps/CABERNET/Functions/Show network dynamical properties`. The dynamical properties analysis is composed by three different tabs, one for each measure. For the attractor length distribution and for the distribution of the basins of attraction it is possible to choose the number of bins to use and the set of networks on which performing the analyses, either the selected networks or all of them. In the rightmost panel some further standard statistics and indexes, such as, e.g., mean, standard deviation, etc. are listed.
It is also possible to export the data as a *.csv file using the *Export CSV file* button [Figure 17].

4.8 Robustness analysis

With CABERNET it is possible to perform a robustness analysis over the simulated networks. To this end, it is necessary to use the *dynamic perturbations* form, accessible from the right-click network menu selecting the `Apps/Perform robustness analysis`

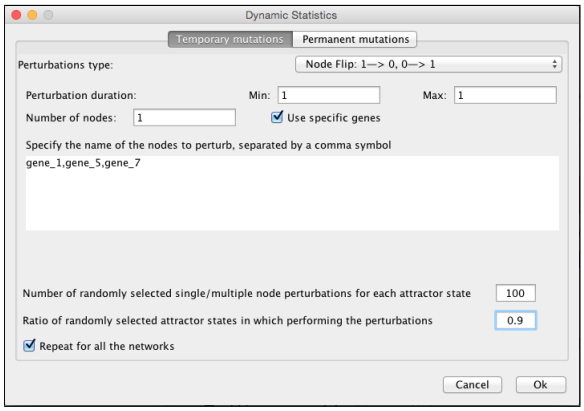

Figure 18: **Robustness analysis:** temporary perturbations tab

`(avalanches and sensitivity)` command or from `Apps/CABERNET/Functions/Perform robustness analysis (avalanches and sensitivity)` [Figure 18].
Two kinds of perturbation experiments are available: (*a*) *Temporary mutations* and (*b*) *Permanent mutations*.

- *Temporary perturbations*. Different parameters are required:

  a) type of perturbation:

  1) flips

  2) knockin/knockout

  b) duration of the perturbation, randomly defined at each experiment according to an uniform distribution probability between the minimum and the maximum values

  c) number of nodes perturbed at each perturbation,

  d) number of states of each attractor in which repeating the perturbation experiment, expressed as ratio of states

  e) number different perturbation experiment. For the knock-in/knock-out type it is necessary to set all the parameters separately. Also, by selecting the *Use specific genes* flag it is possible to set the specific nodes to perturb in a comma-separated names list.

- *Permanent perturbations*. In this case the user can :

  1) set the nodes to knock-in or knock-out listing their names, in a comma-separated list, in the correct box.

  2) choose a number of randomly chosen nodes to perturb.

Note that both the perturbation types, temporary and permanent, are performed at the same time in each experiment. When the computation will be completed, a the dynamic perturbation analysis view will be shown.
The robustness analysis can be performed for the selected network or for all the generated networks (Repeat for all the networks checkbox selected).

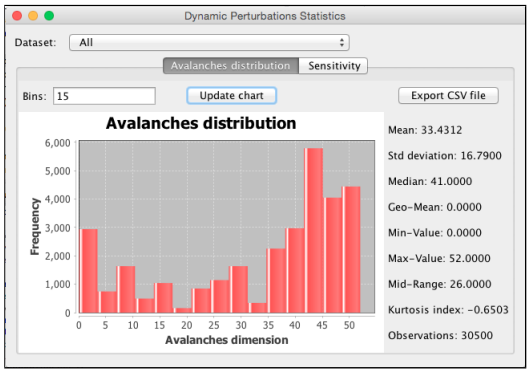

Figure 19: **Robustness analysis report:** avalanches distribution tab

4.8.1 Robustness analysis report: avalanches and sensitivity

This view allows the user to visualize two types of distributions: avalanches and sensitivity [Figure 19].
(*a*) An avalanche is defined as the number of nodes whose activation pattern in the attractor of the wild type network is different if compared to the pattern in the attractor of the perturbed one, in a specific perturbation experiment.
(*b*) The sensitivity of a node is the number of times, calculated for all the experiments, where the status of that node in the original attractor is different if compared to its status in the attractor reached through the perturbation. This view is shown only if the avalanches and the sensitivity have been computed (see Robustness analysis on section 4.8).
The view is composed by two tabs: one for the avalanches distribution and one for the sensitivity. For the avalanches chart it is possible to choose the number of bins and to update the chart by pressing the *Update chart* button. It is possible to choose if compute the chart for only one network or for all the generated networks (Note that this second feature is possible only if the robustness analysis has been performed for all the networks). In the rightmost panel in both the tabs some statistics and indexes, such as, e.g., mean, standard deviation are listed. Also in this case it is possible to export the distributions data as *.csv files.

Appendix: main algorithms

Algorithm 1 Erdös-Rényi and Barabasi-Albert network generation algorithms.

Erdös-Rényi Algorithm
Input: number of nodes v, number of edges e;
set $V \leftarrow \{1, 2, ..., v\}$ and $E \leftarrow \emptyset$;
for $i : 1 \rightarrow e$ **do**
  let $a \sim U[1, v]$ and $b \sim U[1, v]$;
  if $(a, b) \notin E$ **then**
    $E \leftarrow E \cup \{(a, b)\}$
  end if
end for.

Barabasi-Albert Algorithm
Input: number of initial/total (n_i/n_f) nodes, average connectivity m;
requires: $m < n_i < n_f$;
set $V \leftarrow \{1, .., n_i\}$ and $E \leftarrow \emptyset$;
for all $a, b \in V$ **do**
  $E \leftarrow E \cup U(\{(a, b)\}, \{(b, a)\})$;
end for
for $a : n_i + 1 \rightarrow n_f$ **do**
  for $i : 1 \rightarrow m$ **do**
    repeat
      let $p_i \leftarrow k_i / \sum_j k_j$ where k_i is the number of connections for node i;
      set $b \leftarrow R(V)_{p_i}$ and $e \leftarrow U(\{(a, b)\}, \{(b, a)\})$;
    until $e \notin E$
    $E \leftarrow E \cup \{e\}$
  end for
  $V \leftarrow V \cup \{i\}$
end for.

Algorithm 2 Search of attractor

build a empty $2^n - 1$ dimensional vector $attractor[0...2^n - 1]$;
loop
  repeat
    let $s \leftarrow random\{0,1\}^n$ be a n dimensional Boolean random vector;
  until $attractor[s] \neq NULL$
  set $currentAttractor \leftarrow null$ and $visited \leftarrow \emptyset$;
  repeat
    set $position[s] \leftarrow count$, $count \leftarrow count + 1$; ;
    set $visited \leftarrow visited \cup \{s\}$ and $s' \leftarrow F(s)$;
    if $s' \in visited$ **then**
      $currentAttractor \leftarrow s'$
    else
      if $attractor[s'] \neq null$ **then**
        $currentAttractor \leftarrow attractor[s']$
      end if
    end if
    if $currentAttractor \neq null$ **then**
      for all $s \in visited$ **do**
        $attractor[s] \leftarrow currentAttractor$
      end for
    else
      $s \leftarrow s'$
    end if
  until $(currentAttractor = NULL)$
end loop

Algorithm 3 Compute the ATM matrix

initialize an empty matrix $m[|A|,|A|] \leftarrow 0$
for all $a \in A$ **do**
  for all $s \in a$ **do**
    for $i : 0 \rightarrow |s|$ **do**
      $s' \leftarrow mutation(s,i)$
      $m[attractor[s], attractor[s']] \leftarrow m[attractor[s], attractor[s']] + 1$
    end for
  end for
end for
{perform normalization of m}

References

[1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland Science, fifth edition edition, 2007.

[2] H. H. Chang, M. Hemberg, M. Barahona, D. E. Ingber, and S. Huang. Transcriptome-wide noise controls lineage choice in mammalian protenitor cells. *Nature*, 453:544–548, 2008.

[3] C. Furusawa and K. Kaneko. Chaotic expression dynamics implies pluripotency: when theory and experiment meet. *Biol. Direct.*, 4:17, 2009.

[4] A. Graudenzi, G. Caravagna, G. De Matteis, and M. Antoniotti. Investigating the relation between stochastic differentiation, homeostasis and clonal expansion in intestinal crypts via multiscale modeling. *PLoS ONE*, (9(5): e97272), 2014.

[5] K. Hayashi, S. M. Lopes, and M. A. Surani. Dynamic equilibrium and heterogeneity of mouse pluripotent stem cells with distinct functional and epigenetic states. *Cell Stem Cell*, 3:391–440, 2008.

[6] M. Hoffman, H. H. Chang, S. Huang, D. E. Ingber, M. Loeffler, and J. Galle. Noise driven stem cell and progenitor population dynamics. *PLoS ONE*, 3:e2922, 2008.

[7] M. Hu, D. Krause, M. Greaves, S. Sharkis, M. Dexter, C. Heyworth, and T. Enver. Multilineage gene expression precedes commitment in the hemopoietic system. *Genes Dev.*, 11:774–785, 1997.

[8] S. Huang. Reprogramming cell fates: reconciling rarity with robustness. *Bioessays*, 31:546–560, 2009.

[9] D.A. Hume. Probability in transcriptional regulation and its implications for leukocyte differentiation and inducible gene expression. *Blood*, 96:2323–2328, 2000.

[10] T. Kalmar, C. Lim, P. Hayward, S. Munñoz-Descalzo, J. Nichols, J. Garcia-Ojalvo, and A. Martinez Arias. Regulated fluctuations in nanog expression mediate cell fate decisions in embryonic stem cells. *PLoS Biol.*, 7:e1000149, 2009.

[11] A. Kashiwagi, I. Urabe, K. Kaneko, and T. Yomo. Adaptive response of a gene network to environmental changes by fitness-induced attractor selection. *PLoS ONE*, 1:e49, 2006.

[12] S.A. Kauffman. Homeostasis and differentiation in random genetic control networks. *Nature*, 224:177, 1969.

[13] S.A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, 22:437–467, 1969.

[14] T.P. Peixoto and B. Drossel. Noise in random boolean networks. *Phys. Rev. E*, 79:036108–17, 2009.

[15] A.S. Ribeiro and S.A. Kauffman. Noisy attractors and ergodic sets in models of gene regulatory networks. *J. Theor. Biol.*, 247:743–755, 2007.

[16] R. Serra, M. Villani, A. Barbieri, S.A. Kauffman, and A. Colacci. On the dynamics of random boolean networks subject to noise: attractors, ergodic sets and cell types. *J. Theor. Biol.*, 265:185–193, 2010.

[17] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramadge, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, 13(11):2498–504, 2003.

[18] M. Villani, A. Barbieri, and R. Serra. A dynamical model of genetic networks for cell differentiation. *PLoS ONE*, 6(3):e17703. doi:10.1371/journal.pone.0017703, 2011.

[19] Kaizhong Zhang, Rick Statman, and Dennis Shasha. On the editing distance between unordered labeled trees. *Information Processing Letters*, 42(3):133 – 139, 1992.