Modeling of Genetic Networks with Boolean functions

1 Boolean Regulatory Networks

The behavior of a genetic network is not completely determined with their graphical representations. For example, from Fig. 1 one can conclude that gene g_1 inhibits gene g_2, but the behavior of g_2 when g_1 is not active is not defined.

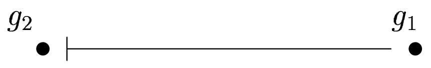

Fig. 1. Inhibition network.

For instance, g_2 can behave "normally" and switch off in the absence of influence by other genes. In the real genetic networks this corresponds to the fact that the protein that is the product of the expression of g_2 is subject to degradation and as such will eventually disappear. Thus, if g_1 is off, regardless of its initial state g_1 will eventually switch off. Alternatively, g_1 can be a self-activator, which in the absence of the inhibition by g_1, eventually switches to an on state. This kind of gene behavior is also realistic - a gene can promote its own transcription in a positive feedback. For instance, the product of the gene is in the same time an activator of the gene transcription.

Another situation when the graphical representation is ambiguous can be seen in Fig. 2.

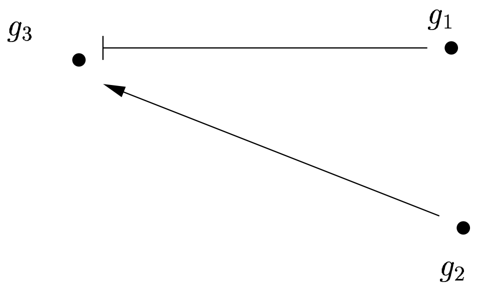

Fig. 2. Ambiguous network.

Gene g_3 is activated by g_1 and in the same time inhibited by g_2. It is not clear which influence will "win", i.e., whether g_3 will go on or off.

We can capture this kind behavior by making a Petri net model of the network, as we saw this during the lectures. As the Petri net models have uniquely defined dynamics we can deduce from it the intended behavior of our genetic network. However, this can be quite a cumbersome way to define the behavior, especially for complex genetic networks that have many elements.

A more direct way to specify genetic networks behavior is by *boolean functions*. Let $B = \{0, 1\}$ the Boolean set that contains two elements: 1 and 0, interpreted, respectively, as the boolean values **true** and **false**, or as the gene being *on* and *off*. Genes from the gene set $G = \{g_1, g_2, \ldots, g_n\}$ of the network can take values from B. A state of the network is given with the valuation function $s : G \rightarrow B$ that assigns a boolean value to each of the genes. The behavior of a given gene g_i is determined by the genes in G. In fact the behavior is affected only by the genes that are connected to g_i and possibly g_i itself, but to keep it simpler, for the time being we will assume that all genes influence g_i. Then formally the new value that g_i will get is given by the boolean function $f : B^n \rightarrow B$ that maps vectors of boolean values (i.e., the states of genes $g_1 \ldots g_n$, including g_i itself) to a boolean value (the new state) of g_i. Thus the function f_i which is associated with g_i completely defines the behavior of g_i. For instance, in the network in Fig. **??** we can express the fact that g_2 spontaneously activates itself when g_2 is absent by defining $f_2(0, 0) = 1$. In words, when g_1 and g_2 are both off g_1 will switch on. Similarly, for the network in Fig. 2 the fact that the inhibition by g_2 wins over the activation by g_1 can be expressed with $f_3(1, 1, 0) = 0$. We define the set $F = \{f_1, f_2\}$ of functions that are associated to the genes.

Of course to fully define the boolean function f_i we will have to assign it a value for each combination of its arguments, i.e., each possible state. This is usually done in a tabular form. Thus, for f_2 corresponding to g_2 in Fig. 1 we can have have the definition given in Tab. 1.

Table 1. Boolean function for gene g_2 in the Inhibition network.

g_1	g_2	$f_2(g_1, g_2)$
0	0	1
0	1	0
1	0	1
1	1	0

Similrly we can define f_3, corresponding to g_3 in Fig. 2, for instance like in Tab. 2:

Although in principle we can associate arbitrary value to a gene, when modeling biological networks, of course, we try to assign values that can be justified with the behavior of the real network.

We assumed for the time being that each gene g_i is influenced by all other genes. So, formally speaking we will had to include among the arguments of function f_i also the genes that according to the graphical representation have

Table 2. Boolean function for gene g_2 in the Inhibition network.

g_1	g_2	g_3	$f_3(g_1, g_2, g_3)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	0	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	0	0

no connection to g_i and therefore no direct influence on its behavior. To avoid this we can further associate to each gene g_i the set $I_i = \{g_{i_1}, g_{i_2}, \ldots, g_{i_k}\}$ of *input* genes that are inputs to g_i. Now we can restrict the definition of f_i only to combinations of values of genes that are in I_i. Note that g_i can also be in I_i in case the new state of g_i depends on the current one. We define the set $I = \{I_1, I_2, \ldots, I_n\}$ as the set that contains all input sets. We can join the genes, their functions and input sets to obtain a new structure (G, I, F) often referred to as *Boolean regulatory graph.* To complete the specification of the system we need to define the transitions between its states.

We assume that the behavior of the network is asynchronous and interleaved. This means that in a given moment of time only one gene can change its state. In other words, there is no true parallelism - no two genes can change simultaneously.There is always an ordering of the changes - they are interleaved. When two or more independent changes are possible then the choice is made non-deterministically. This is consistent with the behavior implied by the Petri-net semantics defined during the lectures.

More formally, like for the Petri nets, we can capture the behavior (semantics) of the network by means of its corresponding transition system (S, S_0, T), where $S \subseteq B^n$ is the finite state of possible states (recall that those are mappings of G to B), set of initial states $S_0 \subseteq S$, and the transition relation $T \subseteq S \times S$. The pair of states (s, s') is in the transition relation T if and only if there exists a gene $g_i(1 \leq i \leq n)$ such that $s'(g_i) = f_i(s(g_{i_1}), s(g_{i_2}), \ldots, s(g_{i_k}))$ and for all $g_j \neq i$ it holds $s'(g_j) = s(g_j)$. In other words, the new state of g_i is obtained exactly via the function f_i applied to the old states of the genes that influence g_i and g_i is the only gene that changes its state.

Further we can restrict ourselves to the states that are reachable from the initial states. A state s' is reachable from a state s iff there exists a sequence of transitions $(s, s_1), (s_1, s_2), \ldots, (s_n, s')$, i.e. a sequence that transforms s into s'. The sequence of transitions can be empty, which means that a state is trivially reachable from itself. Having defined reachability, we can restrict our state space to the part that is reachable from the initial state. We define the set of reachable states $S_r = \{s | s \text{ is reachable from } s_0 \in S_0\}$. Similarly the transition relation can be restricted to $T_r = T \cap (S_r \times S_r)$, which gives us the transition system (S_r, S_0, T_r) which can be much smaller than the original one.

2 Examples

We can always translate Boolean regulatory graphs to Petri nets of the type we considered during the lectures, i.e., Petri nets that have two places per gene and exactly one token in only one of these places. A formal translation can be found in [1]. Conversely, each such Petri net can be translated into a boolean regulatory graph. In what follows we give the Boolean regulatory graph versions of some genetic networks that we modeled previously with Petri nets. Note again that the functions f_i are defined only on genes that are in the input set I_i.

Consider first the inhibition network from Fig 3. The Petri net implementation too is given in Fig. 3. The implementation fully determines the behavior: for instance, one can see that if g_1 is not inhibited by g_2, i.e., $g2 = 0$, then g_1 will spontaneously activate itself.

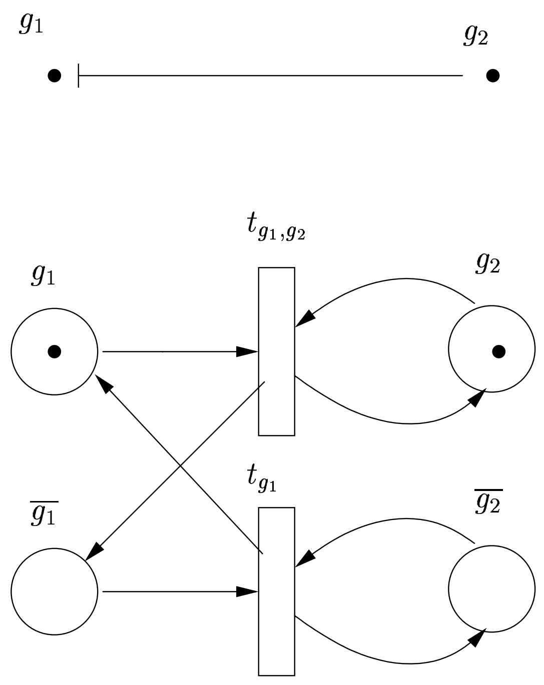

Fig. 3. Inhibition: Network and Petri net versions.

The corresponding boolean regulatory network (G, I, F) is given by

- $G = \{g_1, g_2\}$

- $I = \{I_1, I_2\}$, where $I_1 = \{g_2\}$ and $I_2 = \emptyset$,
- $F = \{f_1, f_2\}$, where $f_1(0) = 1$,$f_1(1) = 0$, and $f_2 = Id$ (Id is the identity function, i.e., g_2 is a constant defined with the initial state.

To completely specify the behavior of the whole network we give the transition system of the network (S, S_0, T). To this end we define the sates $s_0 = (0,0), s_1 = (0,1), s_2 = (1,0), s_3 = (1,1)$, where the values in the binary vector correspond to the genes g_1 and g_2, respectively.

- $S = \{s_0, s_1, s_2, s_3\}$
- $S_0 = S$
- $T = \{(s_0, s_2), (s_3, s_1)\}$

Like in the case of Petri nets we can represent the transition system with a graph that has the states as nodes and the transitions as edges, which is left as an exercise to the reader.

Our second example is the negative feedback genetic network given in Fig. 4.

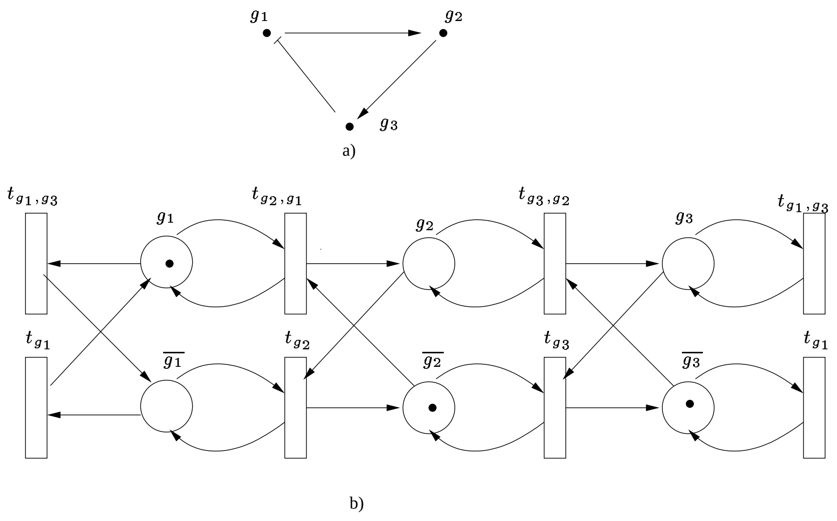

Fig. 4. Negative feedback circuit.

The corresponding Boolean regulatory network (G, I, F) is defined as

- $G = \{g_1, g_2, g_3\}$
- $I = \{I_1, I_2, I_3\}$, where $I_1 = \{g_3\}$, $I_2 = \{g_1\}$, and $I_3 = \{g_1\}$.
- $F = \{f_1, f_2, f_3\}$, where $f_1(0) = 1$,$f_1(1) = 0$,$f_2(0) = (0)$,$(f_2(1) = 1$,$(f_3(0) = 0$, and $f(_3(1) = 1$.

To fully specify the behavior the corresponding transition system (S, S_0, T) should be defined. Analogously to the previous example, let us define the states the sates $s_0 = (0,0,0)$, $s_4 = (1,0,0)$, $s_6 = (1,1,0)$, $s_7 = (1,1,1)$, $s_3 = (0,1,1)$,$s_1 = (0,0,1)$,$s_2 = (0,1,0)$,$s_5 = (1,0,1)$ where the values in the binary vector correspond to the genes g_1, g_2, and g_3, respectively.

Let us assume that the set of initial states is given as $S_0 = \{s_0, s_2, s_5\}$. Then all above defined states s_i are reachable from some initial state and therefore, using the denotation defined in the previous section, $S = S_r = \{s_i | 0 \leq i \leq 7\}$. The transition relation is given as
$T = T_r = \{(s_0, s_4), (s_4, s_6), (s_6, s_7), (s_7, s_3), (s_3, s_1), (s_1, s_0)\}$.

The negative feedback network exhibits an oscillatory behavior and therefore the graph representing the state transition system contains (is) a cycle. Drawing the graph is again left as an exercise to the reader.

References

1. C. Chaouiya, E. Remy, P. Ruet, D. Thieffry, *Qualitative Modelling of Genetic Networks: From Logical Regulatory Graphs to Standard Petri Nets*, Proc. of ICATPN, Lecture Notes in Computer Science 3099, pp. 135-156, Springer-Verlag, 2004.