Composition of Boolean functions restricts biologically permitted logics

Thomas M. A. Fink*† and Ryan Hannam*
*London Institute for Mathematical Sciences, Royal Institution, 21 Albermarle St, London W1S 4BS, UK
†Bit.bio, Dorothy Hodgkin Building, Babraham Research Campus, Cambridge CB22 3FH, UK

We show that composing Boolean functions severely restricts their range of computation. For bipartite systems, where two species depend on each other but not themselves, this heavily constrains the observed behaviour of each species. We apply our insights to gene regulation, where genes interact via transcription factors but only gene-gene interactions are observed. We derive an expression for the number of distinct Boolean functions under composition and show that the fraction of permitted biological logics tends to be very small. We confirm our results with computational enumeration.

Here is a simple question with a surprising answer. Imagine that people only have two moods, happy or sad. As a man, your own mood depends on the moods of two women. For instance, you might be happy only if both women are happy. Or you might ignore them both and always be sad. The mood of each woman depends, in turn, on the mood of two different pairs of men. So, ultimately, your mood is governed by those of the four men. In how many ways can your mood depend on them?

You might guess that there are $2^{2^4} = 65{,}536$ ways, which is the number of logical dependencies on four variables. But in reality there are just 520 ways to depend on the four men. The hidden variables of the women greatly reduces the range of logical dependencies.

The solution to this puzzle hints at a fundamental aspect of dynamical systems in which two species depend on each other but not themselves. It suggests that the logical dependencies observed between a single species are highly restricted. The preeminent example of such a system is genetic regulatory networks, in which genes interact via transcription factors. As we shall see, the range of permissible gene-gene logics is severely constrained.

For 50 years, single-species Boolean networks have been extensively studied as crude models of gene regulatory networks. The original N-K model was an attempt to model cell states [1], in which attractors in the dynamics mirror different cell types [2]. Despite their simplicity, a theoretical understanding of the dynamics of Boolean networks proved elusive until the mid-2000s [3–7].

However, this model has a major drawback: it assumes that just one species of player is involved, when in reality there are two key species. Through a series of biochemical events known as gene expression, genes produce proteins. Some of these proteins, called transcription factors, bind to the DNA and regulate the transcription of genes. In this way, the expression levels of genes are determined by those of other genes, but only indirectly—transcription factors act as middlemen [8].

To address this drawback, a new model of gene regulatory networks has emerged that explicitly accounts for the transcription factor middlemen: bipartite Boolean networks [9–11]. These are Boolean networks in which two species of nodes depend on each other but not themselves. In our social network analogy, men and women play the role of genes and transcription factors. Whether a gene is expressed is a Boolean function of its transcription factor regulators, and whether a transcription factor is synthesized is a Boolean function of its contributing genes.

Bipartite models of regulation can reflect biologically important details, such as different gene and transcription factor connectivities [9, 10] and the effects of activators and inhibitors on steady state dynamics [9]. They are also used to study gene knock-out experiments [11].

So why have these more realistic models taken so long to emerge? One reason is that they are ostensibly harder to study. Another is that, despite the underlying bipartite biochemistry, experimentalists persist in studying networks of gene expression and protein interactions separately [12–14]. In other words, what actually takes place are interactions between genes and transcription factors, but what gets measured are interactions between genes. Our goal is to provide a framework for translating between these perspectives, summarized in Fig. 1.

In this Letter we do three things. First, we show that a bipartite Boolean network can be decomposed into two ordinary Boolean networks, one for each species of nodes. The dynamics of each species is the same in both perspectives. Second, we derive an exact expression for the number of biologically permitted logics, which is the number

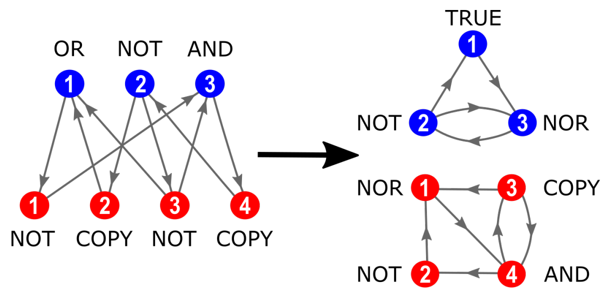


FIG. 1: Bipartite Boolean networks, in which two species depend on each other but not themselves, can be decomposed into two ordinary Boolean networks. The dynamics of each species are identical in both perspectives. The Boolean functions on the right are compositions of those on the left.

of Boolean functions that can be expressed as a composition of Boolean functions. We confirm our prediction with computational enumeration. Third, we apply our insights to genetic regulatory networks which are bipartite over genes and transcription factors.

Throughout this Letter, we refer to Boolean functions and logics interchangeably.

A bipartite BN is two ordinary BNs

This Letter is about dynamics on bipartite networks, which has received little attention [9–11]. But the structure of bipartite networks has been heavily investigated. The structure problem is to determine the connectivity of a single species of nodes from the connectivity between the two species of nodes, where two nodes of the same species are taken to be connected if they are second-nearest neighbors. In our social network analogy, this is finding the connectivity of men with common female friends. This process is called projection, an example of which can be seen in Fig.1, ignoring the Boolean functions. In particular, researchers have focused on the degree distributions of projected networks [15, 16], differences between operators on bipartite graphs and their projections [17], the reduction of the number of cliques on projections due to edge weights [18, 19], and community detection in bipartite networks [20].

Despite the interest in projecting bipartite structure, projecting bipartite Boolean networks dynamics has not been studied. Here we show that a bipartite Boolean network can be decomposed into two ordinary Boolean networks. But unlike the projection of structure, where structure information gets lost in the process, remarkably the projection of dynamics is lossless. In other words, each species will follow identical trajectories in the original and decomposed networks.

Consider a bipartite Boolean network with p nodes of one species and q nodes of another species. We call the two sets of nodes X and Y. The binary states of X and Y are given by $\mathbf{x} = (x_1, \dots, x_p)$ and $\mathbf{y} = (y_1, \dots, y_q)$. Associated with each species is a Boolean function, one for each node: $\mathbf{f} = (f_1, \dots, f_p)$ and $\mathbf{g} = (g_1, \dots, g_q)$. To be clear, $\mathbf{f}$ and $\mathbf{g}$ are associated with X and Y but depend on Y and X. The two state vectors $\mathbf{x}$ and $\mathbf{y}$ change with time according to the Boolean functions:

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{y}(t)) \quad \text{and} \quad \mathbf{y}(t+1) = \mathbf{g}(\mathbf{x}(t)).$$

Combining these,

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{g}(\mathbf{x}(t-1))) = \mathbf{h}(\mathbf{x}(t-1)),$$

where $\mathbf{h} = (h_1, \dots, h_p)$ are the composed Boolean functions. (There will also be another set of compositions for when $\mathbf{f}$ and $\mathbf{g}$ are swapped, but the same arguments apply in both cases.) In other words, the state vector of one species is uniquely determined by its state vector two times steps back, without the need to consider the state vector of the other species. Fig. 1 gives a simple example of this.

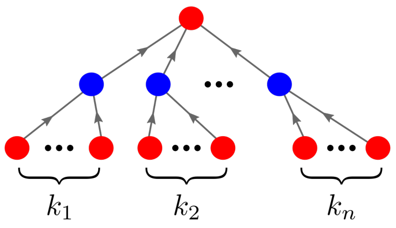


FIG. 2: Schematic representation of some Boolean function f of n variables, each of which is a Boolean function $g_1, \dots, g_n$ of $t_1, \dots, t_n$ variables. We can express this composition using the shorthand $(t_1, t_2, \dots, t_n)$, which we call the composition structure.

Now let's zoom in on the composed Boolean functions $\mathbf{h} = \mathbf{f}(\mathbf{g})$. The fs and gs will depend on a subset of the states $y_1, \dots, y_q$ and $x_1, \dots, x_p$—just which ones depending on the network connectivity. Consider the state of a node x^* with in-degree n. It will be governed by some Boolean function f of n input variables, each of which is a Boolean function g_i with t_i input variables. This composed Boolean function h is a function of $t_1 + \ldots + t_n$ variables, as shown in Fig. 2:

$$\begin{aligned} &h(x^1_1, \dots, x^1_{t_1}; \dots; x^n_1, \dots, x^n_{t_n}) = \\ &f\big(g_1(x^1_1, \dots, x^1_{t_1}), \dots, g_n(x^n_1, \dots, x^n_{t_n})\big). \end{aligned} \tag{1}$$

Here we have assumed that all of the nodes x^j_i are distinct, that is, there are no loops of size two in the bipartite network. We can express such a composition using the shorthand

$$\{t_1, t_2, \dots, t_n\},$$

which we call the composition structure and use throughout what follows. For example, $\{2, 2\}$ is shorthand for $h(x^1_1, x^1_2, x^2_1, x^2_2) = f(g_1(x^1_1, x^1_2), g_2(x^2_1, x^2_2))$.

The main goal of this Letter is to calculate the number of biologically permitted logics, that is, the number of Boolean functions that can be expressed as a composition of Boolean functions, as in eq. (1). As we shall see, it tends to be considerably smaller than the number of ways of assigning Boolean functions to f and $g_1, \dots, g_n$, namely, $2^{2^n} 2^{2^{t_1} + \ldots + 2^{t_n}}$. This in turn tends to be much smaller than the number of Boolean functions of the $t_1 + \ldots + t_n$ variables on which h depends, namely, $2^{2^{t_1} \ldots 2^{t_n}}$.

Number of compositions

Before we calculate the number of distinct compositions of Boolean functions, we review some general properties of them. There are 2^{2^n} Boolean functions of n variables. For $n = 2$, they are true, false, a, b, $\overline{a}$, $\overline{b}$, ab, $a\overline{b}$, $\overline{a}b$, $\overline{a}\overline{b}$, $a+b$, $a+\overline{b}$, $\overline{a}+b$, $\overline{a}+\overline{b}$, $ab+\overline{a}\overline{b}$ and $a\overline{b}+\overline{a}b$. In this notation, $\overline{a}$ means NOT a, ab means a AND b, and $a+b$ means a OR b. Notice that two of these functions depend on no variables (true and false), four depend on one variable (a, b,

$\overline{a}$ and $\overline{b}$), and the rest depend on two variables. Let $a(n)$ be the number of Boolean functions of n variables that depend on all n variables. By the principle of inclusion and exclusion,

$$a(n) = \sum_{i=0}^{n} (-1)^{n-i} \binom{n}{i} 2^{2^i}.$$

The first several $a(n)$ are 2, 2, 10, 218, 64594 (OEIS A000371 [21]).

Generalizing this, let $b(n, m)$ be the number of Boolean functions of n variables that depend on $m \leq n$ variables. Since there are $\binom{n}{m}$ ways of selecting those m variables,

$$a(n, m) = \binom{n}{m} a(m).$$

The first several $a(n, m)$ are

$$\begin{array}{lllll} 2; \\ 2, & 2; \\ 2, & 4, & 10; \\ 2, & 6, & 30, & 218; \\ 2, & 8, & 60, & 872, & 64594. \end{array}$$

Note that $a(n, n) = a(n)$, $a(n, 0) = 2$, and summing $a(n, m)$ over m gives 2^{2^n}.

Let's now calculate the number of distinct compositions. We refer to the interior upstairs Boolean function as f, and the interior ones as $g_1, \ldots, g_n$, as shown in Fig. 2. Let $\gamma(t_1, \ldots, t_n)$ be the number of Boolean functions that depend on at least one variable in each of the $g_1, \ldots, g_n$. (We drop the brackets around $\{t_1, \ldots, t_n\}$ inside functions for convenience.) Let

$$\alpha_n = (2^{2^n} - 2)/2,$$

The quantity α_n is the number of Boolean functions of n variables that depend on at least one of those variables, divided by two to account for inversions. Then

$$\gamma(t_1, \ldots, t_n) = a(n)\, \alpha_{t_1} \ldots \alpha_{t_n}. \qquad (2)$$

For example,

$$\begin{aligned} \gamma(i) &= 2\, \alpha_i, \\ \gamma(i, j) &= 10\, \alpha_i \alpha_j, \\ \gamma(i, j, k) &= 218\, \alpha_i \alpha_j \alpha_k. \end{aligned}$$

Now let $c(t_1, \ldots, t_n)$ denote the number of distinct Boolean functions of n inputs, which are themselves Boolean functions of $t_1, \ldots, t_n$ inputs. This is what we are ultimately interested in calculating. To do so, we simply need to sum γ over the ways of depending on none of the interior Boolean functions g_i, plus the ways of depending on one of the g_i, and so on, up to the ways of depending on all n of the g_i. Then

$$c(t_1, \ldots, t_n) = \sum_{e \in 2^{\{t_1, \ldots, t_n\}}} \gamma(e), \qquad (3)$$

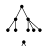
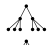
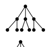
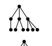
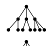
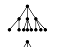
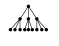
Composition structure $(t_1, \ldots, t_n)$		*Composable logics* $c(t_1, \ldots, t_n)$	*Possible logics* $2^{2^{t_1+\ldots+t_n}}$	*Composable fraction*
1, 1		16	16	1
1, 2		88	256	0.34
2, 2		520	65,536	0.0079
1, 3		1528	65,536	0.023
2, 3		9160	4.3×10^9	2.1×10^{-6}
3, 3		161,800	1.8×10^{19}	8.8×10^{-15}
1, 1, 1		256	256	1
1, 1, 2		1696	65,536	0.026
1, 2, 2		11,344	4.3×10^9	2.6×10^{-6}
1, 1, 3		30,496	4.3×10^9	7.1×10^{-6}
2, 2, 2		76,288	1.8×10^{19}	4.1×10^{-15}
1, 2, 3		204,304	1.8×10^{19}	1.1×10^{-14}
2, 2, 3		1,375,168	3.4×10^{38}	4.0×10^{-33}
2, 3, 3		24,792,448	1.2×10^{77}	2.1×10^{-60}
3, 3, 3		447,032,128	1.3×10^{154}	3.3×10^{-146}

TABLE I: The first column shows, notationally and graphically, the composition structure $\{t_1, \ldots, t_n\}$, which indicates a Boolean function of n inputs, which are themselves Boolean functions of $t_1, \ldots, t_n$ inputs. The second column shows the number of composable Boolean functions. The third shows the number of possible Boolean functions of $t_1 + \ldots + t_n$ variables. The fourth shows the ratio of the composable and possible functions, which is very small for most structures. We tested our predictions against computational enumeration for all but the last three rows and found exact agreement.

where the sum is over the power set of $\{t_1, \ldots, t_n\}$—all subsets e of the set $\{t_1, \ldots, t_n\}$—and written $2^{\{t_1, \ldots, t_n\}}$. For example,

$$\begin{aligned} c(i) &= \gamma(\emptyset) + \gamma(i), \\ c(i, j) &= \gamma(\emptyset) + \gamma(i) + \gamma(j) + \gamma(i, j), \\ c(i, j, k) &= \gamma(\emptyset) + \gamma(i) + \gamma(j) + \gamma(k) \\ &+ \gamma(i, j) + \gamma(j, k) + \gamma(i, k) + \gamma(i, j, k). \end{aligned}$$

Inserting (2) into (3),

$$c(t_1, \ldots, t_n) = \sum_{e \in 2^{\{t_1, \ldots, t_n\}}} a(|e|)\, \alpha_{t_1} \ldots \alpha_{t_{|e|}}.$$

Grouping together subsets of the same size, this becomes

$$c(t_1, \ldots, t_n) = \sum_{m=0}^{n} a(m) \sum_{\sigma_1 \ldots \sigma_m} \alpha_{t_1} \ldots \alpha_{t_n},$$

where the $\sigma_1 \ldots \sigma_m$ are all of the subsets of size m of $\{t_1, \ldots, t_n\}$. For $m = 0$, the sum is over the null set.

This is our main analytic result, and it can be used to calculate the exact number of distinct Boolean functions for any composition. For example,

$$\begin{aligned} c(i) &= 2 + 2\,\alpha_i, \\ c(i,j) &= 2 + 2(\alpha_i + \alpha_j) + 10\,\alpha_i\alpha_j, \\ c(i,j,k) &= 2 + 2(\alpha_i + \alpha_j + \alpha_k) \\ &+ 10\big(\alpha_i\alpha_j + \alpha_j\alpha_k + \alpha_i\alpha_k\big) + 218\,\alpha_i\alpha_j\alpha_k, \end{aligned}$$

where $\alpha_n = (2^{2^n} - 2)/2$. Explicit values of these are given in Table I for i, j and k ranging from 1 to 3. For the case of $c(2,2)$, the 520 Boolean functions are indicated explicitly in Table II. Note in particular that

$$c(t_1, \ldots, t_n) \leq 2^{2^n} 2^{2^{t_1} + \ldots + 2^{t_n}} \leq 2^{2^{t_1} \ldots 2^{t_n}}.$$

The middle term is the number of ways of assigning logics to f and $g_1, \ldots, g_n$, and the right term is the number of logics of $t_1 + \ldots + t_n$ variables.

For composition structures in which all of the inputs are functions of the same number k of inputs, the form of c simplifies:

$$c(k, \ldots, k) = \sum_{m=0}^{n} a(m) \binom{n}{m} \alpha_k^m.$$

For $k = 1$, this reduces to $c(1, \ldots, 1) = 2^{2^n}$, since $a(n)\binom{n}{m} = a(n,m)$ and $a(n,m)$ sums to 2^{2^n}. So for structures such as $\{1,1\}$, $\{1,1,1\}$, the number of composable and possible logics are the same, as shown in Table I.

To test our predictions, we wrote a Mathematica program to compose all possible Boolean functions of all possible Boolean functions for a given composition structure. Specifically, we tested of all but the last three rows in Table I. (The last three rows take to long to compute.) The numbers of distinct composed Boolean functions are in exact agreement with our predictions.

Discussion and application to gene regulation.

We know that the number of permissible logics is drastically reduced when we compose them. But how much intuition do we have for which ones are valid? Table II summarizes all of the valid logics for the composition structure $\{2,2\}$: a Boolean function of two inputs, each of which is itself a Boolean function of two inputs. This is equivalent to the social network analogy in our opening puzzle. Out of the possible $2^{2^4} = 65{,}536$ Boolean functions of four variables, only 520 are valid.

A great deal of this reduction arises from an inability to distinguish between inputs from the same interior function: the $g_1, \ldots, g_n$ in eq. (1). For a composed logic to be valid, it must be invariant over permutations of variables in the same branch. For example, if inputs a and b come from the same branch, interchanging them should not alter the function: $f(g_1(a,b,\ldots),\ldots)$ and $f(g_1(b,a,\ldots),\ldots)$ must be the same.

Since the landmark recognition of induced pluripotent stem cells, there has been a series of discoveries of small numbers of transcription factors which control cell identity. These have the potential for manufacturing cells for personalized and regenerative medicine [22, 23], drug development [24] and disease modelling [25]. The key to reprogramming experiments is identifying combinations of transcription factors which alter the gene expression profile of a cell from one attractor to another. The perturbations are applied by introducing additional, exogenous, copies of the genes which code for those transcription factors. Our insights demonstrate that an individual gene's range of logical dependence on the expression levels of other genes is greatly reduced by the transcription factor middlemen when they are synthesized from multiple genes. This suggests it is more efficient to directly manipulate the transcription factors in the proteome than to do so indirectly via introduction of exogenous genes.

This Letter motivates a number of extensions and open questions. First, we showed that bipartite Boolean networks can be modeled with ordinary Boolean networks. For example, a $K = 2$ random bipartite network projects onto two $K = 4$ random ordinary networks, but with a weighted distribution of the 520 logics shown in Table II. This means that questions about the bipartite dynamics can be answered by studying ordinary dynamics, such as the conditions for the critical type II dynamics which separate order from chaos [].

Second, while we showed that different assignments of Boolean functions map to the same Boolean function under composition, we have not calculated the degeneracy of this many-to-one map. Our observations suggest that some logics show up much more frequently than others, with the most frequent ones tending to depend on fewer variables. For example, for the $\{2,2\}$ composition in Table II, the functions in the left columns tend to be more degenerate than the ones on the right. This would imply that biologically permitted logics are not only restricted, but also tend to be simple.

Third, such a non-uniform degeneracy makes the composition of Boolean networks a preeminent testbed for understanding input-output maps [28]. Many input-output maps in mathematics and nature are biased towards simple outputs, and our model might be amenable to a mathematical explanation for why.

Fourth, we only studied Boolean function composition over two levels, but it may be possible to generalize our results to multiple levels. This could give theoretical backing to computational insights into the robustness and evolvability of electronic circuitry [29]. If the number of composition levels becomes large, the restriction and degeneracy compositions could shed light on the space of functions in some types of neural networks [30].

This research was supported by a grant from Bit.Bio. T. F. acknowledges hospitality from the Bit.Bio laboratory. Both authors acknowledge Andriy Fedosyeyev and Alexander Mozeika for helpful discussions.

0 var.	*1 var.*	*2 variables*	*3 variables*	*4 variables*	*4 variables (cont.)*
$1 \times$ T	$2 \times a$	$4 \times ab$	$8 \times abc$	$16 \times abcd$	$16 \times ab + c + d$
$1 \times$ F		$4 \times a + b$	$8 \times a + b + c$	$16 \times ab + cd$	$16 \times a + b + c + d$
		$2 \times ab + \bar{a}\bar{b}$	$8 \times ab + c$	$16 \times abc + abd$	$16 \times ac + ad + bc + bd$
			$8 \times ac + bc$	$16 \times acd + bcd$	$16 \times abc + abd + acd + bcd$
			$8 \times ab + \bar{a}\bar{b} + c$	$16 \times abcd + \bar{a}\bar{b}\bar{c}\bar{d}$	$16 \times abc + \bar{a}bc + abd + \bar{a}bd$
			$4 \times abc + \bar{a}\bar{b}c$	$16 \times a + b + cd$	$16 \times ac + ad + bc + bd + \bar{a}\bar{b}\bar{c}\bar{d}$
			$4 \times abc + \bar{a}\bar{c} + \bar{b}\bar{c}$	$8 \times ab + \bar{a}\bar{b} + cd$	$8 \times ab + cd + \bar{c}\bar{d}$
			$4 \times ac + \bar{a}\bar{b}\bar{c} + bc$	$8 \times ab + cd + \bar{c}\bar{d}$	$8 \times ab + \bar{a}\bar{b} + c + d$
			$2 \times abc + a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c$	$8 \times a + b + cd + \bar{c}\bar{d}$	
				$8 \times abcd + \bar{a}\bar{b}\bar{c}\bar{d} + abc + abd + \bar{a}bc + \bar{a}bd$	
				$8 \times abcd + \bar{a}\bar{b}\bar{c}\bar{d} + acd + bcd + \bar{a}cd + \bar{b}cd$	
				$4 \times ab + \bar{a}\bar{b} + cd + \bar{c}\bar{d}$	
				$4 \times abcd + ab\bar{c}\bar{d} + \bar{a}\bar{b}cd + \bar{a}\bar{b}\bar{c}\bar{d}$	
				$2 \times abc\bar{d} + ab\bar{c}d + a\bar{b}cd + \bar{a}bcd + a\bar{b}\bar{c}\bar{d} + \bar{a}b\bar{c}\bar{d} + \bar{a}\bar{b}c\bar{d} + \bar{a}\bar{b}\bar{c}d$	

TABLE II: Of the $2^{2^4} = 65{,}536$ Boolean functions of four variables $(f(a,b,c,d))$, only 520 can be expressed as the composition of a Boolean function of two 2-input Boolean functions $(f(g(a,b),h(c,d)))$. In our notation, ab means a AND b, $a+b$ means a OR b, and $\bar{a}$ means NOT a. The columns show the Boolean functions that depend on $m = 0, 1, 2, 3$ and 4 variables. The number of ways to choose those variables is $\binom{4}{m}$. There are $\binom{4}{2}$ choices of two variables, for instance, but we only show the functions for a and b. The number before each function is its inversion degeneracy: the number of functions when none or some of its variables are everywhere replaced by its inverse. For example, the inversions of ab are $ab, a\bar{b}, \bar{a}b$ and $\bar{a}\bar{b}$. The column sums are 2, 2, 10, 50 and 250, and $2\binom{4}{0} + 2\binom{4}{1} + 10\binom{4}{2} + 50\binom{4}{3} + 250\binom{4}{4} = c(2,2) = 520$. Table I gives other composition structure totals.

[1] S. A. Kauffman, Metabolic stability and epigenesis in randomly constructed genetic nets, J Theor Biol **22**, 437 (1969).

[2] S. Huang, G. Eichler, Y. Bar-Yam, and D. E. Ingber, Cell fates as high-dimensional attractor states of a complex gene regulatory network, Phys Rev Lett **94**, 128701 (2005).

[3] S. Bilke and F. Sjunnesson, Stability of the Kauffman model, Phys Rev E **65**, 016129 (2001).

[4] J. E. Socolar and S. A. Kauffman, Scaling in ordered and critical random Boolean networks, Phys Rev Lett **90**, 068702 (2003).

[5] B. Samuelsson and C. Troein, Superpolynomial growth in the number of attractors in Kauffman networks, Phys Rev Lett **90**, 098701 (2003).

[6] I. Shmulevich and S. A. Kauffman, Activities and sensitivities in Boolean network models, Phys Rev Lett **93**, 048701 (2004).

[7] T. Mihaljev and B. Drossel, Scaling in a general class of critical random Boolean networks, Phys Rev E **74**, 046101 (2006).

[8] C. Buccitelli and M. Selbach, mRNAs, proteins and the emerging principles of gene expression control, Nat Rev Genet **21**, 630 (2020).

[9] R. Hannam, R. K?uhn, and A. Annibale, Percolation in bipartite Boolean networks and its role in sustaining life, J Phys A **52**, 334002 (2019).

[10] R. Hannam, Cell states, fates and reprogramming, Ph.D. thesis, King's College London (2019).

[11] G. Torrisi, R. K?uhn, and A. Annibale, Percolation on the gene regulatory network, J Stat Mech **2020**, 083501 (2020).

[12] D. Gomez-Cabrero, et al. Data integration in the era of omics: current and future challenges, BMC Syst Biol **8** I1 (2014).

[13] Y. Liu, A. Beyer, and R. Aebersold, On the Dependency of Cellular Protein Levels on mRNA Abundance, Cell **165**, 535 (2016).

[14] K. J. Karczewski and M. P. Snyder, Integrative omics for health and disease, Nat Rev Genet **19**, 299 (2018).

[15] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Random graphs with arbitrary degree distributions and their applications, Phys Rev E **64**, 026118 (2001).

[16] D. Vasques Filho and D. R. O?Neale, Degree distributions of bipartite networks and their projections, Phys Rev E **98**, 022307 (2018).

[17] K. A. Zweig and M. Kaufmann, A systematic approach to the one-mode projection of bipartite graphs, Soc Netw Anal Min **1**, 187 (2011).

[18] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, Bipartite network projection and personal recommendation, Phys Rev E **76**, 046115 (2007).

[19] Y. Fan, M. Li, P. Zhang, J. Wu, and Z. Di, The effect of weight on community structure of networks, Physica A **378**, 583 (2007).

[20] D. Melamed, Community structures in bipartite networks: A dual-projection approach, PLOS One **9**, e97823 (2014).

[21] N. J. A. Sloane, editor, The On-Line Encyclopedia of Integer Sequences, published electronically at https://oeis.org, 2021.

[22] A. B. C. Cherry and G. Q. Daley, Reprogramming Cellular Identity for Regenerative Medicine, Cell **148**, 1110 (2012).

[23] A. B. C. Cherry and G. Q. Daley, Reprogrammed cells for disease modeling and regenerative medicine, Annu Rev Med **64**, 277 (2013).

[24] S. J. Engle and D. Puppala, Integrating human pluripotent stem cells into drug development, Cell Stem Cell **12**, 669 (2013).

[25] R. R. Kanherkar, N. Bhatia-Dey, E. Makarev, and A. B. Csoka, Cellular reprogramming for understanding and treating human disease, Front Cell Dev Biol **2**, 1 (2014).

[26] J. L. Payne and A. Wagner, Mechanisms of mutational robustness in transcriptional regulation, Front Genet **6**, 322 (2015).

[27] S. E. Ahnert and T. M. A. Fink, Form and function in gene regulatory networks J Roy Soc Interface **13**, 20160179 (2016).

[28] K. Dingle, C. Q. Camargo, and A. A. Louis, Input-output maps are strongly biased towards simple outputs, Nat Commun **9**, 761 (2018).

[29] K. Raman and A. Wagner, The evolvability of programmable hardware, J Roy Soc Interface **8**, 269 (2011).

[30] A. Mozeika, B. Li, and D. Saad, The Space of Functions Computed By Deep Layered Machines, Phys Rev Lett **125**, 168301 (2020).