Boolean composition restricts biological 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

Networks of gene regulation determine cell identity and regulate cell function, but little is known about which logics are biologically favored. We show that, remarkably, the number of logical dependencies that a gene can have on others is severely restricted. This is because genes interact via transcription factors but only gene-gene interactions are observed. We enumerate the number of biologically permitted logics by mapping the problem onto the composition of Boolean functions, and confirm our predictions computationally. This is a key insight into how information is processed at the genetic level.

Puzzle. 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 men (Fig. 1a). So, ultimately, your mood is governed by the mood of the four men. In how many ways can your mood depend on them?

Solution. 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.

Why we care. 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 gene regulatory networks, in which genes interact via transcription factors. As we shall see, the range of permissible gene-gene logics is severely reduced.

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

Drawbacks. 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].

Bipartite Boolean networks. 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. The expression of a gene is a Boolean function of its transcription factor regulators, and the synthesis of a transcription factor 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]. They are also used to study gene knock-out experiments [11].

Two perspectives. 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, an example of which is shown in Fig. 1c.

In this paper. In this paper we do two things. First, we show that a bipartite Boolean network can be projected into two ordinary Boolean networks, one for each species of nodes. The dynamics of a given species is the same in the bipartite and projected perspectives. Second, we derive an exact expression for the number of biologically permitted logics, which is the number of Boolean functions that can be expressed as a composition of Boolean functions. We find that biological logics are severely restricted, and confirm our predictions with computational enumeration. Throughout this paper, we refer to Boolean functions and logics interchangeably.

Projection of bipartite Boolean networks

Network projection. This paper 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

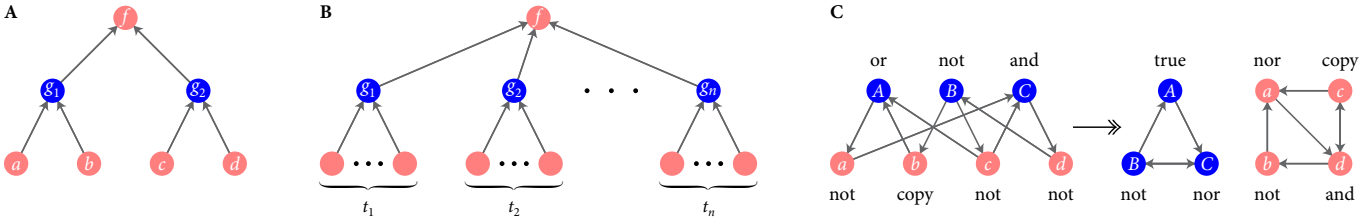

FIG. 1: **Bipartite Boolean networks. A** A Boolean function of two variables, each of which is a Boolean function of two variables. We indicate this composition structure by the shorthand $\{2,2\}$. The function f ultimately depends on a, b, c and d, but the range of logics is highly restricted. **B** A generalization of the composition structure to the left. Now f is a Boolean function of n variables, each of which is a Boolean function of $t_1, t_2, \ldots, t_n$ variables, respectively. We indicate this by $\{t_1, t_2, \ldots, t_n\}$. **C** A bipartite Boolean network, in which two species depend on each other but not themselves. It can be projected 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.

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.1c. Researchers have focused on the degree distributions of projected networks [15, 16] and differences between operators on bipartite graphs and their projections [17], among other things.

Projecting dynamics. Despite the interest in projecting bipartite structure, projecting bipartite dynamics has not been studied. Here we show that a bipartite Boolean network can be projected 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. Each species will follow identical trajectories in the original and projected networks.

Proof of projection. To prove this, 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 $\mathbf{x} = (x_1, \ldots, x_p)$ and $\mathbf{y} = (y_1, \ldots, y_q)$. Associated with each species is a set of Boolean functions, one for each node: $\mathbf{f} = (f_1, \ldots, f_p)$ and $\mathbf{g} = (g_1, \ldots, g_q)$. 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 $\mathbf{h} = (h_1, \ldots, h_p)$ 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 other words, the state vector of one species is uniquely determined by its state vector two steps back, without the need to consider the state vector of the other species. Fig. 1c shows an example of this.

Definition of composition. 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 $y_1, \ldots, y_q$ and $x_1, \ldots, x_p$—just which ones depending on the network connectivity. Consider the state of some node x with in-degree n. It will be governed by its Boolean function f of n input variables, each of which is a Boolean function g_i of t_i input variables. This composed Boolean function h is a function of $t_1 + \ldots + t_n$ variables:

$$\begin{aligned} &h(x_1^1, \ldots, x_{t_1}^1; \ldots; x_1^n, \ldots, x_{t_n}^n) = \\ &f\big(g_1(x_1^1, \ldots, x_{t_1}^1), \ldots, g_n(x_1^n, \ldots, x_{t_n}^n)\big). \end{aligned} \tag{1}$$

We assume that all of the nodes x_i^j are distinct, that is, there are no loops of size four in the bipartite network. We can express such a composition using the shorthand

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

which we call the composition structure. For example, $\{2,2\}$ is shorthand for $h(x_1^1, x_2^1; x_1^2, x_2^2) = f(g_1(x_1^1, x_2^1), g_2(x_1^2, x_2^2))$.

Number of permitted logics

Review of Boolean functions. Before we calculate the number of biologically permitted logics, we review some general properties of logics. (We refer to logics and Boolean functions interchangeably.) There are 2^{2^n} Boolean functions of n variables. For $n = 2$, they are true, false, a, b, $\bar{a}$, $\bar{b}$, ab, $a\bar{b}$, $\bar{a}b$, $\bar{a}\bar{b}$, $a+b$, $a+\bar{b}$, $\bar{a}+b$, $\bar{a}+\bar{b}$, $ab+\bar{a}\bar{b}$ and $a\bar{b}+\bar{a}b$. In this notation, $\bar{a}$ 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, $\bar{a}$ and $\bar{b}$), 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 [19]), starting at $n = 0$. Generalizing this, let

$a(n,m)$ 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

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

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

Two simple bounds. We want to count the number of distinct Boolean functions of the form $f(g_1,\ldots,g_n)$, where the g_i are Boolean functions of $t_1,\ldots,t_n$ inputs, respectively (Fig. 1b), when we run through all possible Boolean functions for f and the g_i. We denote this number by $c(t_1,\ldots,t_n)$, where we drop the braces around $\{t_1,\ldots,t_n\}$ inside functions for convenience. It follows from the definition of c that

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

where the right side is the number of assignments of logics to f and $g_1,\ldots,g_n$. It also follows that

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

where the right side is the number of possible logics of $t_1+\ldots+t_n$ variables. These upper bounds are shown in Table I for n and the t_i ranging from 1 to 3.

Calculating q. Now let's calculate our main result. Let $q(t_1,\ldots,t_n)$ be the number of distinct compositions that depend on at least one variable in each and every of the g_i. The number of choices of g_i that depend on at least one of its t_i variables is $2^{2^{t_i}} - 2$, since true and false depend on no variables. But because both g_i and $\overline{g}_i$ can appear in the main function f and are always distinct, to avoid double counting we must divide this number by two. Let

$$\alpha_{t_i} = (2^{2^{t_i}} - 2)/2.$$

Then

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

For example,

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

Composition structure $\{t_1,\ldots,t_n\}$		*Biological logics* $c(t_1,\ldots,t_n)$	*Assignments of logics* $2^{2^n}2^{2^{t_1}}\ldots 2^{2^{t_n}}$	*Possible logics* $2^{2^{t_1+\ldots+t_n}}$
$\{1,1\}$		16	256	16
$\{1,2\}$		88	1,024	256
$\{1,3\}$		1528	16,384	65,536
$\{2,2\}$		520	4,096	65,536
$\{2,3\}$		9160	65,536	4.3×10^9
$\{3,3\}$		161,800	1,048,576	1.8×10^{19}
$\{1,1,1\}$		256	16,384	256
$\{1,1,2\}$		1696	65,536	65,536
$\{1,1,3\}$		30,496	1,048,576	4.3×10^9
$\{1,2,2\}$		11,344	262,144	4.3×10^9
$\{1,2,3\}$		204,304	4,194,304	1.8×10^{19}
$\{1,3,3\}$		3,680,464	67,108,864	1.8×10^{19}
$\{2,2,2\}$		76,288	1,048,576	1.8×10^{19}
$\{2,2,3\}$		1,375,168	16,777,216	3.4×10^{38}
$\{2,3,3\}$		24,792,448	268,435,456	1.2×10^{77}
$\{3,3,3\}$		447,032,128	4,294,967,296	1.3×10^{154}

TABLE I: **Number of Boolean functions, or logics, for different composition structures.** The composition structure $\{t_1,\ldots,t_n\}$ indicates a Boolean function of n inputs, which are themselves Boolean functions of $t_1,\ldots,t_n$ inputs, respectively. The number of biological logics is the number of logics that can be composed in this way. We compare this to the number of was of assigning logics to f and the g_i, and to the number of possible logics of $t_1+\ldots+t_n$ variables. We tested our predictions against computational enumeration for all but the last three rows and found exact agreement.

We take $q(\emptyset)$ to be $a(0)$, which is 2.

Calculating c. To calculate the number of distinct compositions $c(t_1,\ldots,t_n)$—which is the number of biologically permitted logics—we just need to sum q over the ways of depending on none of the 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. We can write this as

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

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

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

Inserting (2) into (3) gives

$$c(t_1,\ldots,t_n) = \sum_{e\in 2^{\{t_1,\ldots,t_n\}}} a(|e|)\,\alpha_{\sigma_1}\ldots\alpha_{\sigma_{|e|}},$$

where the σ_i are the elements of e. Grouping together subsets of the same size, this becomes

$$c(t_1,\ldots,t_n) = \sum_{m=0}^{n} a(m) \sum_{\sigma_1\ldots\sigma_m} \alpha_{\sigma_1}\ldots\alpha_{\sigma_m}, \qquad (4)$$

where the second sum is over all of the subsets of size m of $\{t_1,\ldots,t_n\}$. For $m=0$, this sum is over the null set and is taken to be 1.

Main result. Eq. (4) is our main result. It can be used to calculate the exact number of distinct Boolean functions for any composition. For example,

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

where $\alpha_i = (2^{2^i}-2)/2$, and so on. 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 permitted Boolean functions are indicated explicitly in Table II.

Testing our theory. To test our predictions, we wrote a computer program to compose all possible logics of all possible logics for a given composition structure. In particular, we tested all of the structures in Table I, apart from the last three which took too long to compute. The numbers of distinct composable logics agree exactly with our predictions.

Uniform structures. For composition structures in which all of the inputs are functions of the same number t of inputs, eq. (4) simplifies:

$$c(t,\ldots,t) = \sum_{m=0}^{n} a(n,m)\alpha_t^m.$$

For $t=1$, this reduces to 2^{2^n}. So for structures such as $\{1,1\}$ and $\{1,1,1\}$, the number of composable and possible logics are the same, as Table I indicates.

Discussion and applications

A remarkable link. We have discovered a remarkable correspondence between biologically permitted logics and the composition of Boolean functions. This is direct consequence of the bipartite nature of the interactions between genes and transcription factors. The range of logical dependence that one gene can have on others is greatly reduced by the transcription factor middlemen. This is a fundamental insight into how networks of gene regulation process information and, ultimately, govern morphogenesis, determine cell identity and regulate cell function.

Example of permitted logics. To gain some intuition for which logics are permitted, we summarize them all for the composition structure $\{2,2\}$ in Table II. This is a Boolean function of two inputs, each of which is a Boolean function of two inputs. It is equivalent to the social network analogy that we started this paper with. Out of the possible $2^{2^4} = 65{,}536$ Boolean functions of four variables, only 520 are biologically permitted.

Biased degeneracy. Our work motivates several applications and extensions, and we present five here. First, while different assignments of logics can map to the same logic under composition, we have not considered here the degeneracies of this many-to-one map. For example, for the composition structure $\{2,2\}$ in Table II, 4,096 assignments map to 520 logics. But the degeneracies are not uniform. Some show up much more frequently than others, with the most frequent logics depending on fewer variables. The logics in the left columns of Table II tend to be more degenerate than those in the right. If, as we believe, this trend persists, it would imply that biological logics are not only restricted, but also tend to be simple.

Input-output maps. Second, the composition of Boolean networks are a preeminent testbed for understanding input-output maps [28]. Many input-output maps in nature and mathematics are many-to-one, but with a non-uniform degeneracy that is biased towards simple outputs. Our composition model might be amenable to a mathematical explanation as to why.

Modelling. Third, since bipartite Boolean networks can be exactly projected onto ordinary Boolean networks, the latter can be used to model the former. For example, a random bipartite network with two inputs at every node projects onto two random ordinary networks with four inputs everywhere, but with a weighted distribution of the 520 permitted logics. Thus questions about bipartite dynamics can be answered by studying ordinary dynamics, such as the conditions for the critical regime separating order and chaos.

Circuits and neural networks. Fourth, we studied Boolean function composition only over two levels, but it should be possible to generalize our results to multiple levels. This could give theoretical backing to computational insights into the robustness and evolvability of circuits [29]. When the number of composition levels is large, this could shed light on the space of functions in some types of neural networks [30].

Cell reprogramming. Fifth, 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 [20, 21]. These have the potential for manufacturing cells for personalized and regenerative medicine [22, 23], drug development [24] and disease modelling [25]. These special combinations of transcription factors give rise to a cascade event which ultimately controls the cell identity. Our insights into biological logics will help make reverse engineering the right transcription factor sets considerably easier.

Acknowledgements. This research was supported by a grant from Bit.Bio. We 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		$4 \times a + b$	$8 \times a + b + c$	$16 \times ab + cd$	$16 \times a + b + c + d$
		$2 \times ab + \overline{a}\overline{b}$	$8 \times ab + c$	$16 \times abc + abd$	$16 \times ac + ad + bc + bd$
			$8 \times ac + bc$	$16 \times acd + bcd$	$16 \times abc + abd + acd + bcd$
			$8 \times ab + \overline{a}\overline{b} + c$	$16 \times abcd + \overline{a}\overline{b}\overline{c}\overline{d}$	$16 \times abc + \overline{a}bc + abd + \overline{a}bd$
			$4 \times abc + \overline{a}\overline{b}c$	$16 \times a + b + cd$	$16 \times ac + ad + bc + bd + \overline{a}\overline{b}\overline{c}\overline{d}$
			$4 \times abc + \overline{a}\overline{c} + \overline{b}\overline{c}$	$8 \times ab + \overline{a}\overline{b} + cd$	$8 \times ab + cd + \overline{c}\overline{d}$
			$4 \times ac + \overline{a}\overline{b}\overline{c} + bc$	$8 \times ab + cd + \overline{c}\overline{d}$	$8 \times ab + \overline{a}\overline{b} + c + d$
			$2 \times abc + a\overline{b}\overline{c} + \overline{a}b\overline{c} + \overline{a}\overline{b}c$	$8 \times a + b + cd + \overline{c}\overline{d}$	
				$8 \times abcd + \overline{a}\overline{b}\overline{c}\overline{d} + abc + abd + \overline{a}bc + \overline{a}bd$	
				$8 \times abcd + \overline{a}\overline{b}\overline{c}\overline{d} + acd + bcd + \overline{a}cd + \overline{b}cd$	
				$4 \times ab + \overline{a}\overline{b} + cd + \overline{c}\overline{d}$	
				$4 \times abcd + ab\overline{c}\overline{d} + \overline{a}\overline{b}cd + \overline{a}\overline{b}\overline{c}\overline{d}$	
				$2 \times abc\overline{d} + ab\overline{c}d + a\overline{b}cd + \overline{a}bcd + a\overline{b}\overline{c}\overline{d} + \overline{a}b\overline{c}\overline{d} + \overline{a}\overline{b}cd + \overline{a}\overline{b}\overline{c}d$	

TABLE II: **Valid logics.** 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 ($f(g(a, b), h(c, d))$). Here ab means a AND b, $a + b$ means a OR b, and $\overline{a}$ 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 $ab, a\overline{b}, \overline{a}b$ and $\overline{a}\overline{b}$. 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. Kühn, 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. Kühn, 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. J. 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] N. J. A. Sloane, editor, The On-Line Encyclopedia of Integer Sequences, published electronically at https://oeis.org, 2021.

[20] M. Pawlowski et al., Inducible and deterministic forward programming of human pluripotent stem cells into neurons, skeletal myocytes, and oligodendrocytes, Stem Cell Reports **8**, 803 (2017).

[21] H. Kamao et al., Characterization of human induced pluripotent stem cell-derived retinal pigment epithelium cell sheets aiming for clinical application, Stem Cell Reports **2**, 205 (2014).

[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).