Dynamical Hierarchy and Modularity in Gene Regulatory Networks

Carlos Rodríguez-Caso
Bernat Corominas-Murtra
Ricard V. Solé

SFI WORKING PAPER: 2008-12-049

SANTA FE INSTITUTE

Dynamical hierarchy and modularity in gene regulatory networks

Carlos Rodríguez-Caso[1], Bernat Corominas-Murtra[1], Ricard V. Solé[1 2]
[1] *ICREA-Complex Systems Lab, Universitat Pompeu Fabra (GRIB-PRBB). Dr Aiguader 88, 08003 Barcelona, Spain and*
[2] *Santa Fe Institute, 1399 Hyde Park Road, Santa Fe NM 87501, USA*

Modularity and hierarchy are two essential traits of biological organization. They pervade the logic of cellular computations, adaptive responses to changing environments and evolvability. However, no general agreement exists on how to properly measure them. Here, we provide a well grounded theoretical definition of dynamical hierarchy and modularity. This is possible through the identification of the dynamical backbone (DB), the minimal subgraph that contains all the dynamically essential components of any gene regulatory network. Our methodology is based on the most elementary trait behind any dynamic behavior: the principle of causality. In gene regulatory networks this principle is captured by the regulatory control of transcription factors on their target genes. As case studies, we analyzed the structure of the DB in both yeast and *E. coli* gene transcriptional regulatory networks. Although these webs display similar global topological patterns, their DBs exhibit dramatically different architectures. A marked top-down hierarchy is present in the *E. coli* net, whereas the yeast network displays a bow-tie structure. Several modules are identified in both systems, although their number and position within the DB is markedly different, suggesting two different forms of logic organization. Our method allows to unambiguously define the core dynamical modules and their hierarchical organization without the use of any tunable parameter and it can be applied to any arbitrary directed graph of causal dependencies.

Keywords: gene regulatory networks, modularity, complex networks, hierarchy, Systems Biology

I. INTRODUCTION

The pattern of regulatory interactions linking transcription factors (TFs) to their target genes constitutes the first level of a multilayered network of gene regulation; the so called gene transcriptional regulatory networks (GRN) [1]. Some of these patterns have been recovered from genome-wide approaches, particularly well established for the bacterium *Escherichia coli* [2–4] and the yeast *Saccharomyces cerevisiae* [5, 6]. In such a picture, both hierarchical [7–9] and modular [6, 10, 11] components have been repeatedly highlighted, although no general agreement exists on what scale of analysis more accurately captures global complexity [12]. Modularity is a widespread, desirable feature of a complex system and is considered a prerequisite for adaptation and evolvability. However, modules are integrated within global networks, often displaying some sort of hierarchical organization. How these two aspects of cellular maps are related is an open problem.

The conceptualization of cellular interaction maps within the framework of graph theory [1, 13, 14] provides powerful insights on their hierarchical [15–18] and modular organization [19–22]. However, their quantification, even their identification has led to a nonuniform concept of module under functional, topological, evolutionary and developmental criteria [23, 24]. Similarly, the observed hierarchy seldom matches an ideal feed-forward relation between components [25]. An alternative approach considers looking at GRNs in a more fundamental way, namely as logic sets of causal relations. Causal links, namely who acts on whom, allows to actually define the skeleton underlying dynamics. In a dynamical setting, the state $\sigma(t) = (\sigma_1(t), ..., \sigma_N(t))$ of a system σ formed by N elements would be updated under some

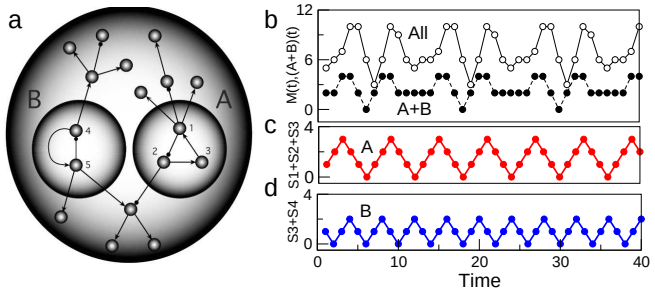

FIG. 1: Using a small threshold network (a) and starting from an initial state where $\sigma_4(0) = \sigma_5(0) = 1$ and $\sigma_i(0) = 0$; $(i \neq 4, 5)$ for other units, a cyclic attractor (of period 12) is obtained. Here only two elements have a nonzero threshold, namely $\theta_1 = \theta_4 = 1/2$. Arrows and end circles indicate $\omega = +1$ and $\omega = -1$ links, respectively. The global state $M(t)$ is given by $M(t) = \sum_{j=1}^{N} \sigma_j(t)$ (thick lines, b). The observed pattern is generated by the activity of two basic feedback loops, indicated as A ad B in (a). These subsets are responsible for the qualitative dynamics exhibited by the net, as shown in (b) with filled circles. Moreover, this attractor results from the combination of the two different periodic orbits displayed by the two basic modules, whose time series are shown in (c-d).

class of dynamical process. An example of such dynamics is a threshold-like equation, namely

$$\sigma_i(t+1) = \Theta\left(\sum_{j=1}^{N} \omega_{ij}\sigma_j(t) - \theta_i\right) \qquad (1)$$

where $\Theta(x) = 1$ if $x > 0$ and zero otherwise. Here θ_i is a threshold and the weights $\omega_{ij} \in \{-1, 0, +1\}$ define the

type of interaction between genes. If the state of each element is Boolean, i. e. $\sigma_i \in \{0,1\}$ the previous model provides, for a given initial state, a closed description of the system's behavior. Here the matrix $W = (\omega_{ij})$ captures the structure and nature of causal links. An example of the resulting dynamics is shown in figure 1, where an ideal GRN is shown (a) where a stable oscillatory cycle is reached (b). We can see that the global pattern for this $N = 16$ network is a periodic cycle, but this pattern is solely due to the behavior of two subgraphs (A and B in figure 1) exhibiting feedback loops. The rest of the system simply reacts to the dynamical inputs generated from these modules. This is true no matter what kind of dynamical rules are used to causally relate our elements. Some elements will play the leading role, determining the qualitative type of dynamics, whereas others will just amplify or reduce the core signals.

This is a computational perspective and not surprisingly, GRNs have been compared to computers [26–28]. Cellular computations pervade both the diverse responses to external stimuli [29, 30] as well as cell robustness and plasticity [31–33]. A number of dynamical approximations suggest a link between network organization and its dynamical behaviour. However, due to the large size of these systems, only a few small systems [34, 35] have been fully analyzed.

Ideally, it would be desirable to have a method to construct a graph capturing all non-trivial causal relations and thus all potentially important computational links. In this context, neither current module detection algorithms nor network dissection in motifs [4, 36] capture such relevant components and all rely on some heuristic approximation. Only in particular cases, such as cell cycle [34], segment polarity in early insect development [37], plant flowering [35] or mesoderm induction [38] where the biological knowledge have permited to reasonably identify the key regulatory interactions a full computational analysis has been performed.

In this paper we show that the limitations of uncovering the dynamical organization of GRNs can be overcome by applying the principle of causality as defined by the directed nature of gene-gene relations. This provides the basis for the relation between dynamics and topology without falling into heuristic or statistical approximations. It permits to define a causal, irreducible core displaying both hierarchical and modular organization of logic units of computation organised in a feed-forward relation order.

II. RESULTS

A. Causality, dynamics and topology

Causal relations in GRNs can be described in terms of directed graphs [1, 13]. A graph $\mathcal{G} \equiv \{V_\mathcal{G}, E_\mathcal{G}\}$ is constituted by a set of vertices or nodes -the genes- $V_\mathcal{G} \equiv \{v_1, ...v_N\}$ and the set of edges linking them - the relations among genes- $E_\mathcal{G} \equiv \{e_1, ..., e_L\}$. The regulatory effect of a TF gene v_i on a specific target gene v_j is captured by the ordered pair $e_k = \langle v_i, v_j \rangle$, depicted by an arrow in the picture of the graph $v_i \rightarrow v_j$. A sequence of vertices $v_1, ..., v_n \in V_\mathcal{G}$ define a *path* in a directed graph $\mathcal{G}$ if

$$(\exists (e_1, ..., e_{n-1}) \in E_\mathcal{G}) : (\forall i < n)(e_i = \langle v_i, v_{i+1} \rangle). \quad (2)$$

If the vertices are genes, a path can be interpreted as a chain of causal relations. We denote a path (if it exists) between v_i and v_j as $\pi(v_1, v_n)$. Interestingly, all TFs exhibit outgoing links, whereas non-TF genes (the target ones) only receive arrows from the TF set. The number of outgoing links of a vertex is known as *out-degree* (denoted by k_{out}) whereas the number of incoming edges is the *in-degree* (k_{in}). Since a TF can be a regulatory target of other TFs, they can exhibit both incoming and outgoing links, allowing feed-backs to occur.

Generically, the pattern of activation of a given gene has a time causal relation with the state of the set of genes affecting it. Indeed, every vertex v_i can acquire a number of possible states $\Sigma(v_i) \equiv \{\sigma_i^1, ...\sigma_i^S\}$. If we define

$$\Gamma_i \equiv \{v_k \in V_\mathcal{G} : \langle v_k, v_i \rangle \in E_\mathcal{G}\} \quad (3)$$

(the set of vertices affecting v_i) the state of v_i at time $(t+1)$, $\sigma_i(t+1)$, is influenced by the state of Γ_i at time t, i.e., $S(\Gamma_i, t)$. Defining $\widehat{S}(\Gamma_i) \supset S(\Gamma_i, t)$ as the set containing the repertoire of all possible $S(\Gamma_i, t)$ configurations, we can define

$$\mathcal{W}_i(\widehat{S}(\Gamma_i)) \longrightarrow \Sigma(v_i) \quad (4)$$

as the set of the correspondences between $\widehat{S}(\Gamma_i)$ inputs and $\Sigma(v_i)$ output states. In this way, every vertex dynamics is obtained from,

$$\sigma_i(t+1) = \mathcal{W}_i(S(\Gamma_i, t)) \quad (5)$$

Here $\sigma_i(t)$ describes the state of a given element at time t. These equations can be formulated in different forms, including Boolean dynamics [39], threshold nets [34] as figure I illustrates or coupled differential equations [40]. These models are different but have a common principle of causality: the state of a given vertex v_i at time $t+1$ is exclusively defined by Γ_i at time t. No matter our choice of the dynamical equations, the patterning of links has a great impact on system's behavior.

The presence of cycles is a necessary (but not sufficient) requisite for a periodic solution. However, the initial state of all components of the cycle influences for its final state. In addition, a cyclic graph implies that every vertex is indirectly affected by itself. As a consequence, its dynamical behaviour cannot be trivially predicted. By contrast, in downstream paths the upstream element fully determines the final network state [properties and formal definition of cycle and linear path are detailed in SI].

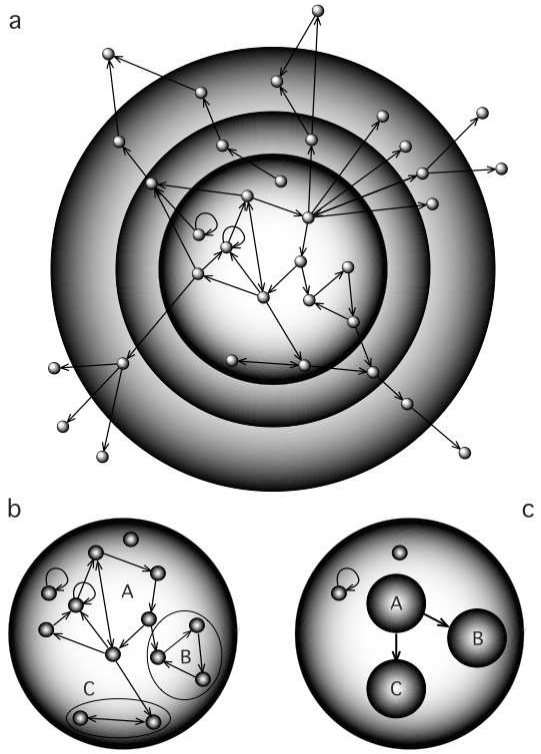

FIG. 2: Dynamical Backbone definition. (a) an example of the iterative elimination of vertices without outdegree ($k_{out} = 0$). Different layers include the remaining set ($B_i = \Theta(B_{i-1})$) of vertices and edges. In (b) we indicate the dynamical modules ($H(\mathcal{G})$ i.e. maximal cycles and isolated root vertices, see text) of the DB. Maximal cycles larger than one vertex are labelled by capital letters. Figure (c) shows the hierarchical organization of the collapsed dynamical modules. Notice that DB exhibits isolated self-interacting vertices that by definition are dynamical modules but auto-loops are maintained in order to provide a more informative picture.

A set able to properly capture the relevant components affecting global behavior should remove linear paths from the graph. Under this view, we can identify this set by means of a straightforward iterative algorithm.

B. Dynamical backbone pruning

We compute the dynamical backbone (DB) of a given directed graph by the iterative pruning of vertices with $k_{out} = 0$ from the initial graph. As we shall see, the DB includes the set of genes determining the qualitative behaviour of the network dynamics. Consider the pruning function $\Phi : \mathcal{G} \to B$, where $B \subseteq \mathcal{G}$. This function takes a directed graph as input and its output is the graph without all the vertices having $k_{out} = 0$ (and the links pointing to them). Accordingly,

$$\Phi(\mathcal{G}) = B_1\{V_{B_1}, V_{E_1}\}, \tag{6}$$

where

$$\begin{cases} V_{B_1} \equiv \{v_k \in V_{\mathcal{G}} : k_{out}(v_k) > 0\} \\ E_{B_1} \equiv \{\langle v_k, v_i\rangle \in E_{\mathcal{G}} : v_k, v_i \in V_{B_1}\}. \end{cases}$$

Thus, the computation is a iterative operation:

$$\begin{aligned} \Phi(\mathcal{G}) &= B_1, \\ B_2 &= \Phi(B_1) \\ &... \\ B_n &= \Phi(B_{n-1}) \end{aligned}$$

The resulting graph at the i-th iteration is denoted by $B_i \equiv B_i(V_{B_i}, E_{B_i})$ and the computation ends when no further vertex elimination occurs, i.e. $B_n = B_{n-1}$. If, for some $i \leqslant n$ a vertex is eliminated and it has no connections with a node belonging to $B_{k \geqslant i}$ we let this vertex *alive*. At every iteration, this collection of *single root* vertices define a set r_i and, from these sets, we build the set R_i of all the single root vertices found until the i-th step:

$$R_i = \bigcup_{k \leqslant i} r_k \tag{7}$$

We have now all the ingredients to provide a formal definition of the dynamical backbone of a directed graph $\mathcal{G}$, $DB(\mathcal{G})$. Let us assume that, when performing recursively the operation Φ over a directed graph, we reached the stable state, i.e., $B_n = B_{n-1}$. The Dynamical Backbone is a subgraph of $\mathcal{G}$, $DB \subseteq \mathcal{G}$ defined as:

$$DB(\mathcal{G}) \equiv DB(V_{B_n}, E_{B_n}) = DB(V_{B_n} \cup R_n, E_{B_n}) \tag{8}$$

Figure (2 a) illustrates the mechanism of pruning. Notice that DB also retains the *single root vertices* i.e. those such that $k_{in} = 0$ which can appear isolated. Root vertices are special in that their state is only externally changed and are not influenced by other genes. We observe that, generally, $DB(\mathcal{G})$ can display more than a single connected component.

Another interesting subgraph $DB'(\mathcal{G})$, is composed by the fraction of the net that exclusively displays feed-forward structures [see S.I.]. To properly identify it, we need to define the subgraph L_n as the set of connections that linking $DB(\mathcal{G})$ to $DB(\mathcal{G})^C$ and the vertices they link. Note that this subgraph may display many components. Its main feature is that the links end in vertices outside the $DB(\mathcal{G})$ but they come from vertices belonging to $DB(\mathcal{G})$. We obtain the feed-forward graph from

$$DB'(\mathcal{G}) = DB(\mathcal{G})^C \cup L_n. \tag{9}$$

The previous definition introduces a drastic reduction of network complexity, as will be shown below in our two case studies.

C. Dynamical Modules and Hierarchy

As we argued above, the qualitative features of the dynamics of the whole net is defined by the DB. Furthermore, the causal relations inside the DB can display

some kind of hierarchy if some subset(s) of nodes affect other subset(s) but not the other way round. Such DB subsets can be explicitly identified and, subsequently, an order relation can be defined among them. This leads to a rigorous definition of Dynamical Hierarchy.

Let us suppose that we are in the k-th connected component of $DB(\mathcal{G})$. The i-th Dynamical Module of the k-th connected component of the $DB(\mathcal{G})$, DM_i^k, is a set of vertices (and the directed edges among them) that constitutes an irreducible unit of causal relations. As we said above, the existence of cycles inside the DB is the responsible of the possible non trivial behavior of the dynamics of the net. Thus, if the k-th component of the DB is not a single root node, the concept of Dynamical Module can be featured with a topological entity that we call *maximal cycle* [See SI for the formal definition]. If $\Delta_k(\mathcal{G})$ is the set of Dynamical Modules of the k-th component of the DB, we can construct another graph,

$$H_k(\mathcal{G}) \equiv H_k(V_{H_k}, E_{H_k}), \tag{10}$$

where $V_{H_k} = \Delta_k(\mathcal{G})$ and E_{H_k} is the set of links connecting the different Dynamical Modules. In other words, we collapse the elements of every Dynamical Module into a single node and we let the links connecting different modules of the component of the DB we are working in. Notice that, as a consequence,

$$H_k(\mathcal{G}) \not\subseteq \mathcal{G}. \tag{11}$$

Interestingly, when we consider these DMs as single vertices of $H_k(\mathcal{G})$, we obtain a feed-forward organized graph. It is precisely the feed-forward organization than enables us to define an order relation among the elements of DB. Such an order relation is straightforwardly interpreted as the dynamical hierarchy of the network's dynamical core and it is defined among the different modules of DB_k. Let $\Pi(H_k(\mathcal{G})) = \{\pi_1^k, .., \pi_m^k\}$ be the set of all existing paths over all nodes of H_k [See SI]. Then, we define the order relation ">" as:

$$(DM_i^k > DM_j^k) \leftrightarrow (\exists \pi(DM_i^k, DM_j^k) \in \Pi(H_k)) \tag{12}$$

The above order relation provides our working definition of dynamical hierarchy.

D. *E. coli* dynamical backbone

We generated the GRN for *E. coli* K-12 prokaryote from the available information in RegulonDB 6.0 database [41]. The resulting network, $\mathcal{G}(V_\mathcal{G}, E_\mathcal{G})$, was a directed graph with $|V_\mathcal{G}| = 1607$ (43 of them with $k_{in} = 0$) and $|E_\mathcal{G}| = 4141$ links with a giant component of 1589 verticesand average degree $\langle k \rangle = 5.1$ [See SI network construction and standard topological analysis]. The network included a total of 156 vertices with $k_{out} > 0$ corresponding with transcription and σ *transacting* factors and 1451 vertices with $k_{out} = 0$, i.e, the target genes.

From *E. coli* GRN we obtained the Dynamical Backbone subgraph, $DB(V_{DB}, E_{DB})$, involving $V_{DB} = 142$ vertices distributed as follows: 33 single root vertices (belonging to the R_n set defined in equation 7) and a set of subgraphs with 109 vertices (10 of them with $k_{in} = 0$) and $|E_{DB}| = 279$ edges. This set is distributed in 9 graphs: a giant component of 100 vertices ($\langle k \rangle = 3.7$), another component displaying two elements and 7 isolated self-interacting vertices. As expected, the genes belonging to DB are described as either transcription or σ factors. Only two exceptions were found: the transcription anti-terminator *cspE* and *trmA*, a tRNA metyltransferase (according to RegulonDB 6.0). It is noteworthy that our topological definition recovers the network obtained by a biological selection criteria considering only trans-regulatory elements [see SI2 for a biological function of DB genes] .

Figure 3a shows the DB organization by collapsing maximal cycles into individual vertices as described earlier. The analysis of the collapsed DB revealed five dynamical modules larger than one vertex (figure 3b). We found that the renormalized DB captures the hierarchical behaviour of the largest graph component evidenced by a feed-forward order relation with six layers of downstream dependencies. By definition, the first layer contains all the vertices with $k_{in} = 0$ but we can see that it also includes the largest dynamical module [figures S2 and S4 of the SI provide a more detailed picture of *E. coli DB*].

The largest module (A in figure 3) contains four of the seven σ factors of *E. coli*. These elements are responsible for transcription initiation. Together with the primary initiator factor *rpoD* (σ^{70}), we find the alternative ones operating under heat shock stress (*rpoH* and *rpoE*, corresponding with σ^{32} and σ^{24}, respectively). In addition, *rpoN* (σ^{54}, initiator of nitrogen metabolism genes) is also part of the module. The second largest hub in the DB is the *crp* gene, also known as CAP (catabolite activator protein). CAP is general regulator that exerts a positive control of many of the catabolite sensitive operons as a sensor of glucose starvation. Other relevant factors are co-localized in this dynamical module as it is the case TFs related with nutrient sensor and assimilation (phosphate sensor system, *phoB*, as well as nucleosid (*cytR*) and arginine (*argP*) transport control. It is noteworthy the initiator factor of DNA replication initiator (*dnaA*) is associated with this group, even more when we also find two specific TF expressed under stress conditions (*lexA* and *cpxR*). Similarly, the other four modules include key genes associated to adaptive responses to changing environmental clues. These include homeostasis in acid environment (stress responses to high osmolarity, module B), antibiotic resistance (lead by *marA* and *marR*, module C), glucitol use (module D) or responses to oxygen changes (module E).

In summary, the *E. coli DB* describes a hierarchical feed-forward network. Within the DB, we identify the dynamical modules responsible for the control of tran-

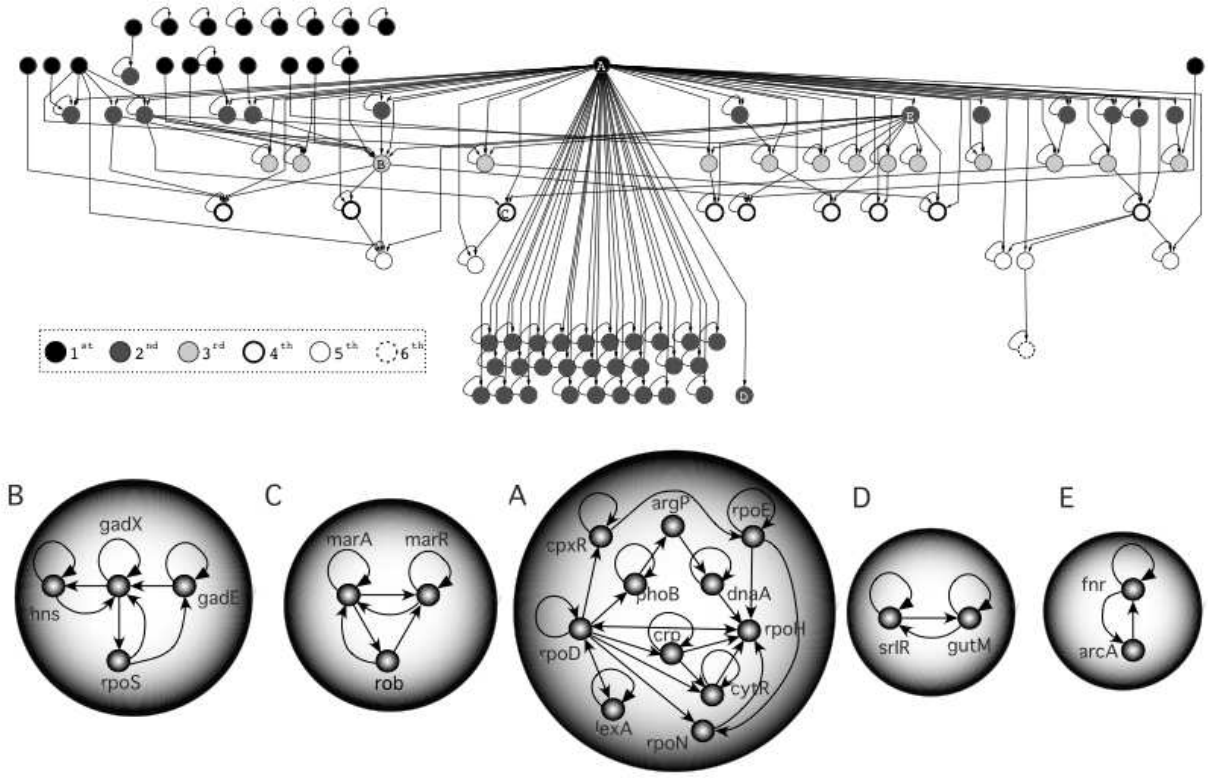

FIG. 3: Dynamical hierarchy and modularity of *E.coli* GRN. Dynamical Backbone of *E.coli* after collapsing dynamical modules revelaing a causal hierarchy (above). Different levels of downstream relations are indicated by means of different vertex labelling. Dynamical modules larger than one vertex are represented by an individual node labelled by capital letters. Auto-links in single-node module are preserved in the graph for a more informative picture. Below *E. coli DB*, the five (A to E) dynamical modules larger than one node are shown.

scriptional replication under both normal and stress conditions, control of metabolism, DNA replication as well as assimilation of essential sources of nitrogen and phosphorous.

E. Yeast dynamical backbone

GRN of *Saccaromyces cerevisiae* eukaryote was obtained from a compilation of different genome scale transcriptional analysis of yeast [6] [see SI for network construction]. The resulting network $\mathcal{G}(V_{\mathcal{G}}, E_{\mathcal{G}})$ was a directed graph consisting of a single connected component with $|V_{\mathcal{G}}| = 4441$ vertices (29 of them with $k_{in} = 0$) and $|E_{\mathcal{G}}| = 12900$ links, leading to $\langle k \rangle = 5.8$ [See SI for standard topological analysis]. The network included a total of 157 vertices with $k_{out} > 0$ and 4284 vertices with $k_{out} = 0$, corresponding with the transcription factors and targets genes analyzed in the different datasets compiled in [6], respectively.

The obtained $DB(V_{DB}, E_{DB})$ was composed by a set of 109 vertices, being 17 of them isolated single root vertices, and a single connected component displaying $|V_{DB}| = 92$ vertices and $|E_{DB}| = 318$ edges, leading to $\langle k \rangle = 6.1$. In spite of *E.coli* and yeast DB exhibited a similar size, they differed in the DB organization. The collapsing process of yeast DB revealed a single dynamical module of 60 vertices (module A in figure 4a) with a high average degree ($\langle k \rangle = 6.6$). The module actually included more than a half of the DB, tied to a much less hierarchical character (see figure 4b) than the observed in the *E. coli DB*. However, the number of layers of computation (seven) was very close in both organisms, as figure 4c indicates.

We can see that the large dynamical module occupies a central position in DB. Overall, the yeast DB resembles the so-called bow-tie organization observed in the Internet [42] where incoming fluxes are integrated in a large component leading to a set of outgoing outputs. This substantially differs from the hierarchical character of *E. coli DB*.

All the 109 genes of the DB appears included in the list obtained from [6] defining the TF genes of the yeast GRN. In addition, we independently checked that, all of DB elements, excluding five, were clearly identified as TFs in the current version Saccaromyces genome

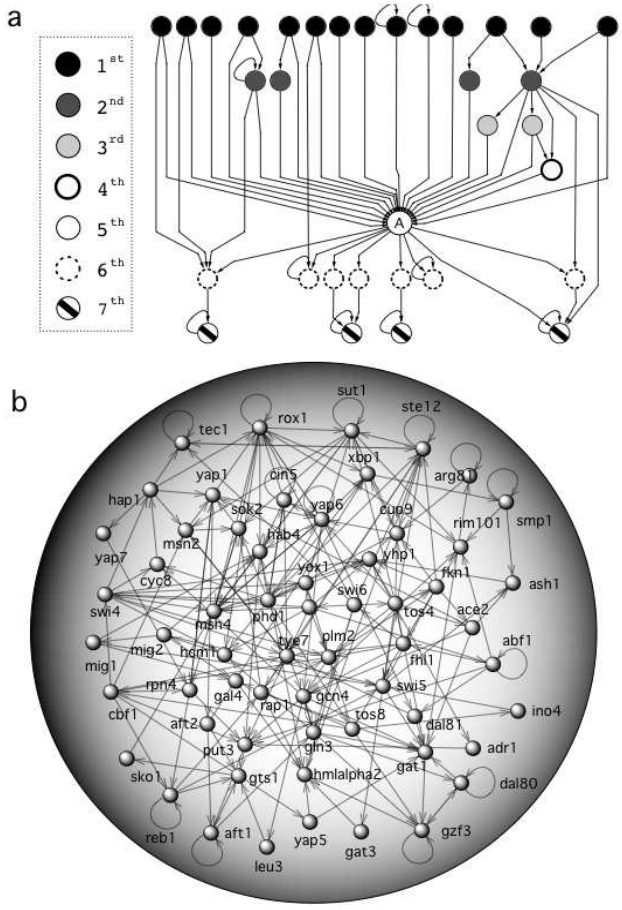

FIG. 4: Dynamical hierarchy and modularity of *Saccaromyces cerevisiae* GRN. Dynamical Backbone of yeast after collapsing dynamical modules (above). One single dynamical module is represented by a vertex labelled by A letter. Below, the A dynamical module. Box legend shows the levels of computation.

database [43]. However, remarkably they all were classified as regulators of gene expression [See SI2 for biological details of DB genes].

Interestingly, module A contains relevant TFs for cell cycle such as *swi4*, *swi6* and *swi5* (they control G_1 related genes and they are also involved with DNA repair), *hcm1* (S phase related genes), *yhp1* and *yox1* (M/G_1 phase) and *fkh* (G_2 phase), among others. Moreover, the swi TFs located in module A make part of the 5 TFs contained in the minimal dynamical network suggested for yeast cell cycle [34]. In spite of such a network combines protein modification besides transcription, we observe the all TFs in Li's network are in yeast DB. The two remaining TFs in Li's module that are not inside module A (*mcm1, mbp*) are located upstream in the hierarchy, indicating a master control over the swi factors and (the remainder module A partners) at transcriptional level [figures S3 and S5 of the SI provide a more detailed picture of yeast DB] . It is also found a number of TF controlling the assimilation of carbon sources, amino acid assimilation (*gal, adr1, aye7, myg1-2, put3, arg81, leu3, gln3* and *gcn4*), nitrogen compound degradation (*dal80-81, gzf3, gat1*) and stress response (*msn4,sut1,yap1-6, sko1, smp1* and *msn2*). A detailed description of biological functions of module A and yeast DB is detailed in SI2.

III. DISCUSSION

While looking at the presence of hierarchical modularity in cellular networks, three basic approaches are followed. The first deals with a statistical analysis of topological patterns at different scales. By looking for well-defined regularities, some important information can be gathered concerning constrains, fragility and efficiency. The topological approach uncovers a first layer of cell complexity and because of its simplicity it provides a powerful, systematic tool of network exploration. The lack of a direct mapping into functionality and the problems arising from the presence of non-adaptive traits [23, 44–46] limits the scope of this view. A different view is a naturalistic approach focusing on functionally meaningful modules playing well-defined roles. Here a detailed knowledge allows the identification of the relevant players and their interactions. Once such modules are characterized as almost-isolated entities, their robustness and evolvability can be studied. Unfortunately, unless such a detailed knowledge is available, no systematic identification of relevant modules is achieved. Finally, a third avenue is given by computational systems biology, using simplified models of regulatory maps. This alternative has been useful in suggesting potential advantages of some given patterns of regulation at different scales, but they require either a previous knowledge of the wiring or some basic assumption of what to search for.

All the previous frameworks define complementary descriptions of GRNs and the final goal is to reach a systematic picture of *what is relevant* and what is not. In this paper we have followed a different path. By using the causality principle, we can systematically reduce network complexity from thousands of elements to a much smaller subset including the *computationally* relevant parts of the GRN. Using such causal/computational perspective is consistent with the essentially computational nature of regulatory maps. No matter how they behave exactly (and thus how they would be modelled) the set of non-reducible causal links should capture what is dynamically relevant. Moreover, our approach does not make any assumption and provides a unique backbone where the presence of a modular hierarchy can be easily depicted.

Our analysis of two well-known GRNs shows that dynamical backbones are very small (compared with the whole network) and thus that the basic logic of GRNs can be strongly simplified. The hierarchical organization of DB for *E. coli* compared with the bow-tie for yeast constitute the most striking difference of these two sys-

tems. The presence of a single, large module in yeast, as oposed to several small ones in *E. coli*, suggests a centrally organized machinery integrating and processing information. In this context, the DB is simpler in *E.coli* although has enriched its dynamical complexity at other scales such as by a more extensive metabolic control by metabolite sensitive TFs or at the level of cell signalling. The simpler pattern displayed by the bacterial DB seems in agreement with its dominant hierarchical structure. The method is formally well grounded and can be applied to other directed graphs involving well-defined causal relations among components.

IV. METHODS

Dynamical backbone algorithm and dynamical module detection were implemented in perl language. Graph pictures were performed using Cytoscape software (http://www.cytoscape.org/).

GRN for *E.coli* was obtained from RegulonDB 6.0 [41] (see S.I for details). Yeast GRN was obtained from the compilation of different sources performed by [47]. Self-interactions initially were not included in that work and they were directly provided by the authors. Data corresponds with highly confident experiments ($P = 0.001$, see [47] for technical details).

Acknowledgments

We thank Complex Systems Lab members for their fruitfull suggestions. This work was funded by the 6th Framework project ComplexDis NEST-043241 (CRC), James McDonnell Foundation (BCM) and the Santa Fe Institute (RVS).

APPENDIX A: DYNAMICS INTO TOPOLOGY

1. GRTNs as directed graphs

GRTNs are formalised in the framework of graph theory as directed graphs. A graph $\mathcal{G} \equiv \{V_\mathcal{G}, E_\mathcal{G}\}$ is constituted by a set of vertices -the genes- $V_\mathcal{G} \equiv \{v_1, ...v_N\}$ and the set of edges linking them - the relations among genes- $E_\mathcal{G} \equiv \{e_1, ..., e_L\}$ [1, 13]. In a directed graph, the edge is an ordered pair $e_k = \langle v_i, v_j\rangle$ depicted by an arrow in the picture of the graph $v_i \rightarrow v_j$. The arrow implies that v_i affects to v_j but not the opposite. Interestingly, since TF genes are regulated too, vertices of this pool can present both incoming and outgoing edges. The number of outgoing links of a vertex is known as *outdegree* (denoted by k_{out}) whereas the number of incoming edges corresponds with *indegree* (k_{in}). Let's define $\Gamma_i \equiv \{v_k \in V_\mathcal{G} : \langle v_k, v_i\rangle \in E_\mathcal{G}\}$ as the set of vertices affecting to v_i.

a. Paths and Cycles

A sequence of vertices $v_1, ..., v_n \in V_\mathcal{G}$ define a *path* in a directed graph $\mathcal{G}$ if:

$$(\exists(e_1, ..., e_{n-1}) \in E_\mathcal{G}) : (\forall i < n)(e_i = \langle v_i, v_{i+1}\rangle). \quad \text{(A1)}$$

We denote a path (if it exists) between v_i and v_j as $\pi(v_1, v_n)$. Note that $\pi(v_i, v_j)$ (if it exists) in $\mathcal{G}$ is itself a subgraph of $\mathcal{G}$, whose set nodes is $V_{\pi(v_i,v_j)} \equiv \{v_i, ..., v_j\}$ and whose set of edges is $E_{\pi(v_i,v_j)} \equiv \{\langle v_i, v_{i+1}\rangle, ..., \langle v_{j-1}, v_j\rangle\}$. We denote the k possible paths (if any) between two nodes as $\pi_1(v_i, v_j), ..., \pi_k(v_i, v_j)$. (Note that the labeling is arbitrary and we can manipulate it to obtain an accurate description). The set of all paths we can define over $\mathcal{G}$ is denoted as $\Pi(\mathcal{G})$. Once we defined a path as a subgraph of $\mathcal{G}$ it is straightforward to define the length of such a path, $l(\pi_k(v_i, v_j))$. Indeed, given two nodes $v_i, v_j \in V_\mathcal{G}$,

$$l(\pi_k(v_i, v_j)) = \begin{cases} |V_{\pi_k(v_i,v_j)}| \leftrightarrow \pi_k(v_i, v_j) \in \Pi(\mathcal{G}) \\ \infty \leftrightarrow \pi_k(v_i, v_j) \notin \Pi(\mathcal{G}) \end{cases} \quad \text{(A2)}$$

(The value ∞ if the path do not exists is just a convention).

As we shall see, the concept of Dynamical Backbone is crucially related with the concept of cycle in a directed graph. We define a *cycle* as a closed loop taking into account the order relation imposed by the arrows. Specifically, we say that a sequence of nodes $v_1, ..., v_n \in V_\mathcal{G}$ belongs to the set of cycles of order n, K_n, of the graph $\mathcal{G}$, if the following relation holds:

$$(v_1, ..., v_n \in K_n) \leftrightarrow [(\exists(e_1, ..., e_n) \in E_\mathcal{G}) : (\forall i < n)(e_i = \langle v_i, v_{i+1}\rangle) \wedge (e_n = \langle v_n, v_1\rangle)] \quad \text{(A3)}$$

(As above, the labeling is arbitrary). Of course, we can accept cases where $n = 1$, when a node is connected with itself.

Notice that the above definition of cycle is compatible with the fact that some fraction of the cycle is repeated in the sequence of nodes, what it would implie that a

cycle is embedded inside another cycle of higher order. In general terms, we talk about *Maximal Cycle* when a cycle is not contained in any other cycle of $\mathcal{G}$. In figure 1 we develop an example that clarifies the concept of maximal cycle.

The concept of path enables us to define an interesting quantity, namely the set of *previous* neighbours of order m of a given node v_i, Γ_i^m:

$$\Gamma_i^m = \{(v_k \in V_{\mathcal{G}}) : (\exists[\pi(v_k, v_i) \in \Pi(\mathcal{G})] \wedge [l(\pi(v_i, v_j)) = m]\} \bigcup \rho_i^m \tag{A4}$$

Where

$$\rho_i^m \;=\; \{(v_k \in V_{\mathcal{G}}) : (\exists \pi_s(v_i, v_k) \in \Pi(\mathcal{G})) \wedge (l(\pi(v_i, v_j)) < m) \wedge k_{in}(v_j) = 0\}$$

The set ρ_i^m contains the roots of all finite paths of length $l < m$ that come through v_i. As a convention, if a given vertex v_k belongs to Γ_i^m and it is also a member of a cycle of order n, then we assume that it will belong to all $\Gamma_i^{m+\beta n}$, being $\beta \in \mathbb{N}$.

To end with, we say that, given two nodes v_i, v_j, $\pi(v_i, v_j) \in \Pi(\mathcal{G})$ is a *linear path* if $\nexists v_k (v_k \neq v_i) \in V_{\pi(v_i, v_j)}$ such that v_k belongs to a cycle. Notice that we allow v_i, the head vertex of the path, either to belong to some cycle or to be a single root vertex.

2. Gene dynamics and computational cost

Gene dynamics can be formalised by a number of approximations, mainly threshold and Boolean networks [39], threshold nets [34] or coupled differential equations [40]. Commonly for all different variations, every vertex v_i can acquire a number of possible states $\Sigma(v_i) \equiv \{\sigma_i^1, ...\sigma_i^S\}$. The state of v_i at time $(t + \Delta t)$, $\sigma_i(t + \Delta t)$ is influenced by the state at time t of Γ_i, i.e., $S(\Gamma_i, t)$. Δt indicates a given minimal timescale of observation and, hereafter $\Delta t = 1$ for simplification notations. Let's define $\widehat{S}(\Gamma_i) \supset S(\Gamma_i)$ as the set containing the repertoire of all possible $S(\Gamma_i)$ configurations. This allows to define $\mathcal{W}_i(\widehat{S}(\Gamma_i)) \longrightarrow \Sigma(v_i,)$ as the set of the correspondences between $\widehat{S}(\Gamma_i)$ inputs and $\Sigma(v_i,)$ output states [52]. In this way, every vertex dynamics is defined as,

$$\sigma_i(t + 1) = \mathcal{W}_i(S(\Gamma_i, t)) \tag{A5}$$

Scaling up to network level, the state configuration at time t for every vertex defines the state of the network $S_{\mathcal{G}}(t) \equiv \{\sigma_1^k, ...\sigma_N^l\}$. From a initial state configuration $S_{\mathcal{G}}(0)$, the iteration of the network rules gives a particular dynamical trajectory that after enough number of iterations (let's say τ) falls in a solution called attractor. We use A_i (A of attractor) to define the set of solutions for one particular trajectory. Among other, more exotic behaviors, the nature of attractors can be punctual (one single solution, i.e. $A_i \equiv S_{\mathcal{G}}(\tau)$ along the time) or conforming a limit cycle (a repeated sequence of $S_{\mathcal{G}}$, i.e. $A \equiv \{S_{\mathcal{G}}(\tau), ..., S_{\mathcal{G}}(\tau + n)\}$ with $(n+1)$ length of period). In this way, the possible attractors and their respective basins of attraction can be recovered by the computation of the trajectories for different initial states. Formally, the landscape of solutions will be given by the exploration of all possible states configurations. However, this is inviable for large networks, since the number possible combinations formally scales as $|\widehat{S}_{\mathcal{G}}| = \prod_i^N \mid \Sigma_i \mid$, where N is the number of vertices of the graph. Indeed, in the simplest case of genes only acquire on/off possible states, an unaffordable computation cost appears since the number of solutions increases by the power of N, namely, $|\widehat{S}_{\mathcal{G}}| = 2^N$. Furthermore, as we will show, the responsible of the whole dynamics of the net could be a small fraction of nodes conforming subgraphs with specific properties.

In order to overcome this limitation we must reduce the number of elements involved in the dynamics but without a loss of complexity understood as the quality and diversity of attractors. Inspired on the target genes of GRNs, we can eliminate those elements that do not qualitatively contribute to dynamical behaviour of the network.

APPENDIX B: DYNAMICAL BACKBONE DEFINITION AND TOPOLOGICAL CONSEQUENCES

In this section we define and develop the concept of Dynamical Backbone and how its topological features have

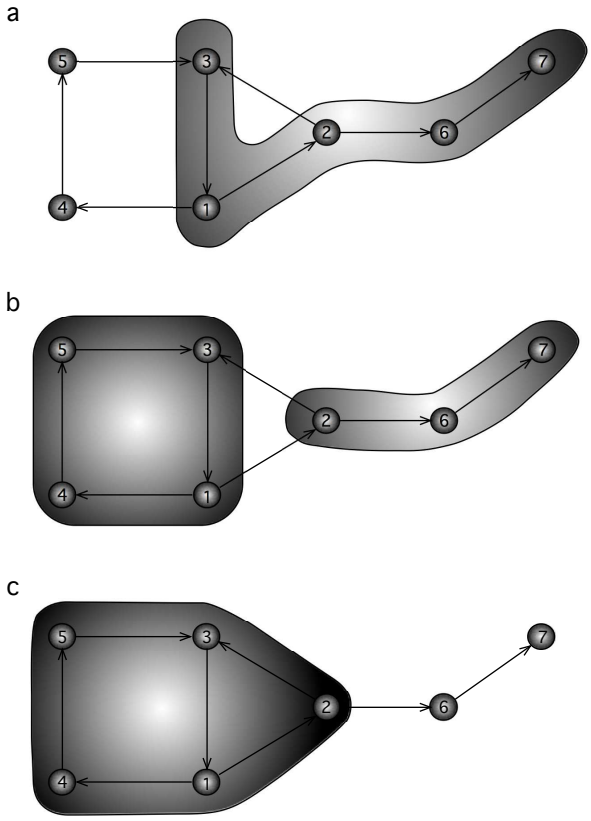

FIG. 5: Representation of a) A path $\pi(v_3, v_7)$, whose set of links is the sequence $E_{\pi(v_3,v_7)} = \{\langle v_3, v_1\rangle, \langle v_1, v_2\rangle, \langle v_2, v_6\rangle, \langle v_6, v_7\rangle\}$. Note that, in this case, there exists only a single path from v_3 to v_7 b) A cycle (left side of the graph) $\mathcal{K} = (\langle v_1, v_4\rangle, \langle v_4, v_5\rangle, \langle v_5, v_3\rangle, \langle v_3, v_1\rangle)$ and a linear path $\pi_q(v_2, v_7)$, such that $E_{\pi_q} = \{\langle v_2, v_6\rangle, \langle v_6, v_7\rangle\}$. Notice that a second cycle (not remarked in the picture) can be found: $\mathcal{K}' = (\langle v_1, v_2\rangle, \langle v_2, v_3\rangle, \langle v_3, v_1\rangle)$. c) illustrates a *maximal cycle* obtained by constructing a cyclic sequence considering all the edges of both $\mathcal{K}$ and $\mathcal{K}'$, namely $(\langle v_1, v_2\rangle, \langle v_2, v_3\rangle, \langle v_3, v_1\rangle, \langle v_1, v_4\rangle, \langle v_4, v_5\rangle, \langle v_5, v_3\rangle, \langle v_3, v_1\rangle)$.

implications for the whole dynamics of a finite and directed net. The mathematics are standard and we enunciate a few, very simple lemmas that enable to grasp the power of the approach.

1. Dynamical Backbone definition

We compute the Dynamical Backbone (DB) of a given directed graph $\mathcal{G}$, $DB(\mathcal{G})$, by the iterative pruning[53] of vertices with $k_{out} = 0$.

Let's define the pruning function $phi : \mathcal{G} \to B$, where $B \subseteq \mathcal{G}$. This function takes a directed graph as input and its output is the graph without all the nodes having $k_{out} = 0$ (and the links pointing to them). Accordingly, $\Phi(\mathcal{G}) = B_1\{V_{B_1}, V_{E_1}\}$ where $V_{B_1} \equiv \{v_k \in V_{\mathcal{G}} : k_{out}(v_k) > 0\}$ and $E_{B_1} \equiv \{\langle v_k, v_i\rangle \in E_{\mathcal{G}} : v_k, v_i \in V_{B_1}\}$. Thus, the computation is a recursive operation like:

$$\begin{aligned} \Phi(\mathcal{G}) &= B_1 \\ B_2 &= \Phi(B_1) \\ &... \\ B_n &= \Phi(B_{n-1}) \end{aligned}$$

The resulting graph at the i-th iteration is denoted by $B_i \equiv B_i(V_{B_i}, E_{B_i})$ and the computation ends at the n-th iteration, being n the smallest positive integer such that $B_n = B_{n-1}$. If at the step $i \leqslant n$ a node is eliminated and it has not any connection with a node belonging to B_k, $k \geqslant i$ we let this node *alive*, although it has no links. At every iteration, this collection of *single root* nodes define a set r_i and, from these sets, we build the set R_i of all the single root nodes found until the step i:

$$R_i = \bigcup_{k \leqslant i} r_k \tag{B1}$$

The reason to conserve such nodes is that, as we should see, they determine the dynamics of some fraction of the net and no other node or set of nodes determine its own dynamics.

We have all the ingredients to define the Dynamical Backbone, of a directed graph $\mathcal{G}$, $DB(\mathcal{G})$. Let us suppose that, when performing recursively the operation Φ over a directed graph, we reached the stable state, i.e., $B_n = B_{n-1}$. $DB(\mathcal{G})$ is a subgraph of $\mathcal{G}$ defined as:

$$DB(\mathcal{G}) \equiv DB(V_{B_n}, E_{B_n})) = DB(V_{B_n} \cup R_n, E_{B_n})) \tag{B2}$$

Thus, at the end, even in the extreme case where $V_{B_n} = \varnothing$, $|V_{DB}| = |R| \geq 1$ since we will have at least, a single root node. Therefore, for any directed graph $\mathcal{G}$

$$DB(\mathcal{G}) \neq \varnothing. \tag{B3}$$

Furthermore, $DB(\mathcal{G})$ can display more than a connected component. The complement set of $B_i \cup R_i$, $(B_i \cup R_i)^C \equiv \{\mathcal{G} \setminus (B_i \cup R_i)\}$ consists of all the removed vertices and edges resulting from the i iterations of Φ.

The set of connections that come from $B_i \cup R_i$ to $(B_i \cup R_i)^C$ define the subgraph L_i. Note that this subgraph may display many components. Its main feature is that the links end in vertices of $(B_i \cup R_i)^C$ but they come from vertices belonging to $B_i \cup R_i$. We formally define this subgraph as:

$$L_i(\mathcal{G}) \equiv L_i(V_{L_i}, E_{L_i}) \tag{B4}$$

where

$$E_{L_i} \equiv \{(\langle v_j, v_k\rangle \in E_{\mathcal{G}}) : (v_j \in (B_i \cup R_i) \wedge (v_k \in (B_i \cup R_i)^C))\} \quad \text{(B5)}$$
$$V_{L_i} \equiv \{(v_k \in V_{\mathcal{G}}) : [(\exists v_j \in V_{\mathcal{G}}) : (\langle v_j, v_k\rangle \in E_{L_i}) \vee (\langle v_k, v_j\rangle \in E_{L_i})]\} \quad \text{(B6)}$$

As we will see later, the resulting graph $DB'_i(\mathcal{G})$

$$DB'_i = (B_i \cup R_i)^C \cup L_i \quad \text{(B7)}$$

captures interesting information about the network dynamics. Accordingly, at the i-th stage of the computation, it is possible to reconstruct the initial graph as:

$$\mathcal{G} \equiv B_i \cup DB'_i \cup R_i. \quad \text{(B8)}$$

2. DB properties

We are ready to proof three lemmas that emphasize the relation with the Dynamical Backbone and the presence of *linear paths* and *cycles* within a directed network. Summarizing, we will see that 1)DB' contains no cycles, 2) If the DB is not a collection of isolated vertices, then it contains, at least, one cycle and 3) all paths start in the DB.

Lemma 1: *Let $\mathcal{G} = \mathcal{G}(E_{\mathcal{G}}, V_{\mathcal{G}})$ be a directed graph. If we apply recursively Φ until the stable state, $DB(\mathcal{G})$, is reached (the Dynamical Backbone), then $DB'(\mathcal{G})$ contains no Cycles.*

Proof: First, note that a vertex $v_k \in DB \cap DB' = L_n$ does not participate in any cycle inside DB' since we only take into account the directed links that connect them with the vertices belonging to $(B_n \cup R_n)^C$ and, inside this subgraph, their $k_{in} = 0$. Furthermore, if $v_k \in R_n$, then, v_k will not participate in any cycle since, by definition, $k_{in} = 0$. Now let us suppose that $v_k \in DB'(\mathcal{G})$ and that there is a sequence $v_1, ..., v_k, ..., v_n \in DB'$ defining a cycle. Thus, there exist a finite i such that:

$$(v_k \in V_{B_{i-1}}) \wedge (v_k \notin V_{B_i})$$

which implies that, at the step $i-1$ of the computation, $k_{out} = 0$. However, if v_k belongs to a cycle and when we apply the operation Φ, it will always display, at least $k_{out} = 1$. Thus it implies that the whole cycle would belong to $DB(\mathcal{G})$, which contradicts the assumptions of the lemma.□

At the other hand, we can see that, except in the trivial cases where $V_{DB(\mathcal{G})} = R_n$, inside the Dynamical Backbone there must be, at least, a cycle:

Lemma 2: *Let $\mathcal{G} = \mathcal{G}(E_{\mathcal{G}}, V_{\mathcal{G}})$ be a directed graph. If we apply recursively Φ until the stable state, $DB(\mathcal{G})$, is reached and $B_n \neq \varnothing$ then $DB(\mathcal{G}) \setminus R$ contains, at least, one cycle.*

Proof: By its definition, when we reached the stable state,

$$(\forall v_k \in V_{DB(\mathcal{G})} \setminus R) \rightarrow (k_{out}(v_k) > 0)$$

Thus it implies that, if we are inside this $DB(\mathcal{G}) \setminus R$ we can always find a path to abandon the vertex we reached and, since the defined net is finite, soon or later we will repeat some fraction of the whole path we did, and this fraction will be a cycle. Note that if this path is an autoloop, we reach again v_k, and it is already a cycle.□

We can conclude that cycles in $\mathcal{G}$ are the responsible of the stopping of the pruning process performed by the recursive application of Φ over $\mathcal{G}$. Furthermore, we can derive another consequence from the definition of the DB and the above properties:

Lemma 3:*Let n be the smallest positive integer such that $B_n = B_{n-1}$. Then, $(\forall v_i \in V_{\mathcal{G}})$ and $(\forall m > n)$;*

$$(v_k \in \Gamma_i^m) \rightarrow (v_k \in V_{DB(\mathcal{G})}) \quad \text{(B9)}$$

Proof: It is easy to see that the series of Γ^i of a given node follow the way by which the pruning algorithm cut the graph until it reaches a stable form. Thus, if the algorithm needed n steps to reach the DB, it implies, by Lemma (1) that the size of the longest path length outside $DB(\mathcal{G})$ is $n-1$. But once we reach the Dynamical backbone, there is no way to climb against the flow of arrows to escape from them, since it would imply that 1) we reached a vertex $v_k \in \rho_i^m$ but not in $DB(\mathcal{G})$, which is a contradiction, or that 2) we reached a cycle where some vertex *receives* an arrow from the outside of the Dynamical Backbone, and we would be again in contradiction with the definition of $DB(\mathcal{G})$. □

Thus, all paths start in the Dynamical Backbone.

3. Dynamical Modules and Hierarchy

The above definitions are valid for every component of DB. Now we will focus on an arbitrary connected component of $DB(\mathcal{G})$, namely, $DB_k(\mathcal{G})$. Given the k-th connected component of $DB(\mathcal{G})$, a Dynamical Module, DM_i^k, is a set vertices (and links) that constitutes an irreducible unit of causal relations. As we will see, this can be featured with the above defined topological entity that we called *maximal cycle*. Interestingly, when we consider this DM's as single vertices, we can define an order relation among the elements of DB which it is straightforwardly interpreted as the dynamical hierarchy of the net.

a. Dynamical Modules

Let us define the i-th Dynamical Module of the k-th component of the DB, DM_i^k; $DM_i^k \subseteq DB_k \subseteq DB(\mathcal{G})$ as

the subgraph whose vertices satisfy the following logical implication[54]:

$$V_{DM_i^k} = \{(v_1, ..., v_m) \in V_{DB_k(\mathcal{G})} : [(\forall v_l, v_s)(\pi(v_s, v_j) \in \Pi(\mathcal{G})) \wedge (\pi(v_j, v_s) \in \Pi(\mathcal{G}))]\}. \quad \text{(B10)}$$

Furthermore, we take as a convention that the members of R are dynamical modules as well as single vertices belonging to the $DB(\mathcal{G})$ but not contained in any cycle. In general terms, a dynamical module is the subgraph defined as:

$$\begin{aligned} DM_i^k &\equiv DM_i^k(V_{DM_i^k}, E_{DM_i^k}) \\ &= DM_i^k(V_{DM_i^k}, \{\langle v_j, v_s\rangle \in E_{\mathcal{G}}) : (v_j, v_s \in V_{DM_i^k})\}) \end{aligned}$$

Recall that, in equation (B10), we are taking into account that we must be able to come back to the vertex we choose to begin the exploration. We temporally refer to the set of all possible dynamical modules of $DB_k \subseteq DB(\mathcal{G})$ as $\Delta_k(\mathcal{G})$. A Dynamical Module $DM_i^k \in \Delta_k(\mathcal{G})$ is called *Maximal* if $(\nexists DM_j^k \in \Delta_k(\mathcal{G})) : (DM_i^k \subset DM_j^k)$. Notice that this definition is equivalent with the definition of maximal cycle, as long as our dynamical modules are not single vertices since they are themselves maximal dynamical modules. Hereafter we will talk only about Maximal Dynamical Modules and, for the sake of simplicity, we will refer simply as *Dynamical Modules*. Consistently, $\Delta_k(\mathcal{G})$ will be the set of Maximal Dynamical Modules of the k-th component of the Dynamical Backbone.

From the definition of the Dynamical Modules of the k-th component of the Dynamical Backbone, we construct another graph $H_k(\mathcal{G}) \nsubseteq \mathcal{G}$,

$$H_k(\mathcal{G}) \equiv H_k(V_{H_k}, E_{H_k}) \equiv H_k\left(\Delta_k, \left[\bigcup_{i \leqslant |\Delta_k|} E_{DM_i^k}\right]^C\right) \quad \text{(B11)}$$

i.e., we collapse the elements of every Dynamical Module into a single node and we let the links connecting different modules. The following lemma is crucial in order to define dynamical hierarchies:

Lemma 4: *Let DB_k be the k-th connected component of the $DB(\mathcal{G})$. Then, $H_k(\mathcal{G})$ contains no cycles.*

Proof: Let us suppose that $H_k(\mathcal{G})$ contains a cycle. Then, by the definition it must be a Dynamical Module and, thus, it will collapse in a single node. This forbids the possibility that $H_k(\mathcal{G})$ contains a cycle.□

b. *Dynamical Hierarchy*

Now we are ready to define *Dynamical Hierarchy* among the different modules of DB_k. Let $\Pi(H_k(\mathcal{G})) = \{\pi_1^k, .., \pi_m^k\}$ be the set of all paths we can define over H_k. Then, we define an order relation ">" as:

$$(DM_i^k > DM_j^k) \leftrightarrow (\exists \pi(DM_i^k, DM_j^k) \in \Pi(H_k)) \quad \text{(B12)}$$

Note that it is possible that not all DM's are comparable within such an order relation. Our definition of order relation is relative to a given path within H_k. Thus, one could ask whether it is possible to find that in some path $DM_i^k > DM_j^k$, but in another path $DM_i^k < DM_j^k$. This is not possible, as we see in the following lemma:

Lemma 5: *Let DB_k be the k-th connected component of the $DB(\mathcal{G})$. Then we define $H_k(\mathcal{G})$ as in (B11) and an order relation as in (B12). If, from a given path π_s we conclude that $(DM_i^k > DM_j^k)$ then, $(\forall \pi_g(DM_i^k, DM_j^j) \in \Pi(H_k))(DM_i^k > DM_j^k)$.*

Proof: We proceed by contradiction, as above. Indeed, the lemma proposes a situation that implies that given $DM_i^k, DM_j^k \in V_{H_k}$, there exist $\pi'(DM_i^k, DM_j^k) \in \Pi(H_k(\mathcal{G}))$ and $\pi'(DM_j^k, DM_i^k) \in \Pi(H_k(\mathcal{G}))$. But it must imply the presence of a cycle, which is not possible by Lemma (4).□

The order relation defined in equation (B12) is the Dynamical Hierarchy of the k-th component of $DB(\mathcal{G})$.

APPENDIX C: DYNAMICAL IMPLICATIONS

In this section we study the dynamical implications of the topological features studied above. As we said in the introductory section, the state of v_i at time $t+1$ is determined by the state of the set Γ_i at time t, i.e., $S(\Gamma_i, t)$ through the function $\mathcal{W}_i$, i.e, $\sigma_i(t+1) = \mathcal{W}_i(S(\Gamma_i, t))$ -eq. (A5).

As an example, let us suppose a linear, finite graph:

$$v_1 \longrightarrow v_2 \longrightarrow, ..., \longrightarrow v_\tau \longrightarrow, ..., \longrightarrow v_n$$

At time t, the state of the node v_τ (located at distance τ from the root node) will be a single, finite, composition

of the functions $\mathcal{W}$ of every node such that,

$$\sigma_\tau(t) = \mathcal{W}_\tau \circ \mathcal{W}_{\tau-1} \circ ... \circ \mathcal{W}_2(\sigma_1(t-\tau)) \tag{C1}$$

Crucially, all nodes except v_1 will not affect *qualitatively* the dynamical behavior of the whole chain. Once $\sigma_1(t)$ is determined, the dynamical behavior of the other nodes is determined by simply, finite, causal relation. The qualitative features of the dynamics will be determined completely by the dynamical behavior of v_1. We can say, without any loss of generality, that $v_2, ..., v_n$ do not affect the qualitative dynamical behavior of the whole chain. The situation changes completely if such set of nodes defines a cycle, since all the nodes of a cycle and its associated Γ^m' will affect the dynamics of the net, maybe leading to an attractor displaying oscillations or some other complex behavior.

In the above example, $\Gamma_k = \{v_{k-1}\}$. However, the interesting cases are those where $|\Gamma_k| > 1$. Let us suppose we have a node v_k such that $v_k \in V_{\mathcal{G}\setminus DB(\mathcal{G})}$. Thus, despite $|\Gamma_k| > 1$, since in $DB'(\mathcal{G})$ there are no cycles (Lemma 1) the qualitative nature of the dynamics will be determined by the qualitative behavior of the nodes belonging to $DB(\mathcal{G})$. In turn, these vertices can be connected in a non-trivial way (maybe defining a cycle) with several vertices belonging to $DB(\mathcal{G})$. Strictly speaking, the qualitative dynamical behavior of a given vertex $v_k \in V_{\mathcal{G}\setminus DB(\mathcal{G})}$ will be determined by the vertices belonging to:

$$(\Gamma_k^m) : (m > n) \tag{C2}$$

which is a set of vertices belonging to $DB(\mathcal{G})$, as demonstrated in Lemma (3).

This implies that the responsible of the whole dynamical behavior of the net is the subgraph defined by $DB(\mathcal{G})$.

Furthermore, given a node $v_k \in V_{\mathcal{G}\setminus DB(\mathcal{G})}$, we can define the set of dynamical modules qualitatively affecting its dynamical state, $\mathcal{D}_k$ i.e,

$$\mathcal{D}_k \equiv \{(DM_i^j \in DB(\mathcal{G})) : ((v_k \in (\Gamma_k^m) : (m > n)) \wedge (v_k \in DM_i^j))\} \tag{C3}$$

Thus we can reproduce the flow of the *signal* and the path through this is processed by the different dynamical units (either dynamical modules or single nodes).

APPENDIX D: CONSTRUCTION OF GRNS

1. E. coli network definition and construction

GRN for E.coli is an overlapping of two files obtained from RegulonDB 6.0 [41]: NetWorkSet.txt, containing TFs and their target genes, and SigmaNetWorkSet.txt, containing the Sigma factors and the genes promoted by them. Both files contain information about the relations, as well as the activator/repressor behaviour, of TF (and Sigma factors) over the target genes. Biological information was obtained from EcoCyc database [50]. TFs and Sigma factors are controlling elements of gene regulation. The main difference is that Sigma factors are essential for basal transcriptional machinery assembly. The participation of one particular Sigma factor is mainly determined by environmental stress and growth phase of the cells, while TFs binds to DNA regulatory region recognition on target genes, promoting or avoiding the transcription initiation. In this work we exclude elements contributing to a TF modification such as phosphorylation or ligand-TF binding. Graph pictures were performed using Cytoscape software (http://www.cytoscape.org/).

2. Topological analysis of *E. coli* GRN

The resulting network is a directed graph of 1607 vertices and 4046 edges, where 87 vertices contain an autolink and 42 display $k_{in} = 0$, whereas the number of vertices with $k_{out} = 0$ (terminal vertices) is 1475. The network is composed by a giant component of 1589 vertices, one of 14 and another one of 4 vertices. The giant component is a *dissasortative* graph ($R = -0.27$) [51] with an average degree $\langle k \rangle = 5.1$. The graph presents a very high average clustering coefficient, $\langle C \rangle = 0.43$, and a lower value of the average shortest path length. $\ell = 2.7$, compared with the prediction for the equivalent Erdös-Rènyi graph model, suggesting that *E. coli* GRN presents a pattern between small and ultra-small world[55]. The degree abundance $v(k)$ was calculated from the giant component of the network. We considered an undirected version of this network. In all cases we used the cumulative abundance, $v_<(k)$. The $v_<(k)$ for the giant component of *E. coli* GRN is an heterogeneous distribution, remaining a power-law behavior ($v_<(k) = 1079.8k^{-1.08}$, $r = 0.976$, where r is, in this case, the linear correlation coefficient). However, it is possible to differentiate two regimes: the first one, for the lowest degrees, with an exponential behaviour ($v(k) = 2787.4e^{-0.45k}$ $r = -0.996$) and the second one following a power law fashion ($v(k) = 484.9k^{-0.89}$ $r = -0.993$) for intermediate and large degrees. The heterogeneity of this distribution can be explained by the fact of we are considering conceptually two different sets of elements and two differentiated reg-

ulatory mechanisms; i.e, the transcription factors that control and are controlled defining a set of regulators, and the target genes that do not participate in the regulation.

Interestingly, clustering distribution as a function of connectivity reveals a power law fashion ($C(k) = 2.22k^{-1.01}$ $r = -0.94$). This dependence has been attributed to a hierarchical organization [17, 49].

3. *S cerevisiae* network definition and construction

Yeast GRN was obtained from the compilation of different sources performed by [47]. Self-interactions initially were not included in that work and they were directly provided by the authors. Data corresponds with highly confident experiments ($P = 0.001$ and three positive replicas). Graph pictures were performed using Cytoscape software (http://www.cytoscape.org/).

4. Topological analysis of *S. cerevisiae* GRN

GRN is a single giant connected component network of 4,441 vertices and 12,864 edges. The graph contains 18 vertices with self-interactions and 29 vertices with $k_{in} = 0$. The graph is dissasortive $R = -0.59$. Its average clustering coefficient $\langle C \rangle = 0.08$ and average shortest path length $\ell = 3.49$ reveal a small world pattern. $v_<(k)$ (cumulative binning) shows a long-tail, heterogeneous shape remaining a power-law $v(k) = 4205.7k^{-1.04}$, $r = 0.94$). Deviations of this behaviour can be attributed to the fact that GRNs are composed by two types of vertices (TF and target genes) with different graph and biological properties.

Clustering coefficient distribution as a function of connectivity reveals that clustering coefficient depends on the degree, but with poorer correlation than the observed for *E. coli* ($C(k) = 0.6k^{-0.96}$ $r = 0.74$)

[1] Babu MM, Luscombe NM, Aravind L, Gerstein M, Teichmann SA (2004) Structure and evolution of transcriptional regulatory networks. Curr Opin Struct Biol 14: 283-291.

[2] Thieffry D, Huerta AM, Pérez-Rueda E, Collado-Vides J (1998) From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in *Escherichia coli*. Bioessays 20: 433-440.

[3] Dobrin R, Beg QK, Barabási, AL, Oltvai ZN (2004) Aggregation of topological motifs in the *Escherichia coli* transcriptional regulatory network. BMC Bioinformatics 5: 10.

[4] Shen-Orr SS, Milo R, Mangan S, Alon U (2002) Network motifs in the transcriptional regulation network of *Escherichia coli*. Nat Genet 31: 64-68.

[5] Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, et al. (2002) Transcriptional regulatory networks in *Saccharomyces cerevisiae*. Science 298: 799-804.

[6] Balaji S, Babu MM, Iyer LM, Luscombe NM, Aravind L (2006) Comprehensive analysis of combinatorial regulation using the transcriptional regulatory network of yeast. J Mol Biol 360: 213-227.

[7] Lagomarsino MC, Jona P, Bassetti B, Isambert H (2007) Hierarchy and feedback in the evolution of the *Escherichia coli* transcription network.Proc Natl Acad Sci U S A 104: 5516-5520.

[8] Yu H Gerstein M (2006) Genomic analysis of the hierarchical structure of regulatory networks. Proc Natl Acad Sci U S A 103: 14724-14731.

[9] Salgado H, Santos-Zavaleta A, Gama-Castro S, Millán-Zárate D, Díaz-Peredo E, et al. (2001) RegulonDB (version 3.2): transcriptional regulation and operon organization in Escherichia coli K-12. Nucleic Acids Res 29: 72-74.

[10] Resendis-Antonio O, Freyre-González JA, Menchaca-Méndez R, Gutiérrez-Ríos RM, Martínez-Antonio A, et al. (2005) Modular analysis of the transcriptional regulatory network of *E. coli*. Trends Genet 21: 16-20.

[11] Wolf DM, Arkin AP (2003) Motifs, modules and games in bacteria. Curr Opin Microbiol 6: 125-134.

[12] Bornholdt S. (2005) Systems biology. Less is more in modeling large genetic networks. Science 310: 449-451.

[13] Albert R (2005) Scale-free networks in cell biology. J Cell Sci 118: 4947-4957.

[14] Zhu X, Gerstein M, Snyder M (2007) Getting connected: analysis and principles of biological networks. Genes Dev 21: 1010-1024.

[15] Clauset A, Moore C, Newman MEJ (2008) Hierarchical structure and the prediction of missing links in networks. Nature 453: 98-101.

[16] Trusina A, Maslov S, Minnhagen P, Sneppen K (2004) Hierarchy measures in complex networks. Phys Rev Lett 92: 178702.

[17] Vázquez A, Pastor-Satorras R, Vespignani A (2002) Large-scale topological and dynamical properties of the Internet. Phys Rev E Stat Nonlin Soft Matter Phys 65: 066130.

[18] Sales-Pardo M, Guimerà R, Moreira, AA, Amaral LAN (2007) Extracting the hierarchical organization of complex systems. Proc Natl Acad Sci U S A 104: 15224-15229.

[19] Ravasz E, Somera AL, Mongru DA, Oltvai ZN, Barabási AL (2002) Hierarchical organization of modularity in metabolic networks. Science 297: 1551-1555.

[20] Rodríguez-Caso C, Medina MA, Solé RV (2005) Topology, tinkering and evolution of the human transcription factor network. FEBS J 272: 6423-6434.

[21] Palla G, Derényi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature 435: 814-818.

[22] Newman MEJ (2004) Fast algorithm for detecting community structure in networks. Phys Rev E Stat Nonlin Soft Matter Phys 69: 066133.

[23] Wagner GP, Pavlicev M, Cheverud, JM (2007) The road to modularity. Nat Rev Genet 8: 921-931.

[24] Hartwell LH, Hopfield JJ, Leibler S, Murray AW (1999) From molecular to modular cell biology. Nature 402: C47-C52.

[25] Whyte LL, Wilson AG, Wilson DM (1969) Hierarchical Structures. New York: Elsevier.

[26] Istrail S, De-Leon SBT, Davidson EH (2007) The regulatory genome and the computer. Dev Biol 310: 187-195.

[27] Bray D (1995) Protein molecules as computational elements in living cells. Nature 376: 307-312.

[28] Macía J, Solé RV (2008) Distributed robustness in cellular networks: insights from synthetic evolved circuits. J R Soc Interface. In press.

[29] Luscombe NM, Babu MM, Yu H, Snyder M, Teichmann SA, Gerstein M (2004) Genomic analysis of regulatory network dynamics reveals large topological changes. Nature 431: 308-312.

[30] Balázsi G, Barabási AL, Oltvai ZN (2005) Topological units of environmental signal processing in the transcriptional regulatory network of *Escherichia coli*. Proc Natl Acad Sci U S A 102: 7841-7846.

[31] Aldana M, Balleza E, Kauffman S, Resendiz O (2007) Robustness and evolvability in genetic regulatory networks. J Theor Biol 245: 433-448.

[32] Lee DS, Rieger H (2007) Comparative study of the transcriptional regulatory networks of *E. coli* and yeast: structural characteristics leading to marginal dynamic stability. J Theor Biol 248: 618–626.

[33] Shmulevich I, Kauffman SA, Aldana M (2005) Eukaryotic cells are dynamically ordered or critical but not chaotic. Proc Natl Acad Sci U S A 102: 13439-13444.

[34] Li F, Long T, Lu Y, Ouyang Q, Tang C (2004) The yeast cell-cycle network is robustly designed. Proc Natl Acad Sci U S A 101: 4781-4786.

[35] Espinosa-Soto C, Padilla-Longoria P, Alvarez-Buylla ER (2004) A gene regulatory network model for cell-fate determination during *Arabidopsis thaliana* flower development that is robust and recovers experimental gene expression profiles. Plant Cell 16: 2923-2939.

[36] Milo R, Shen-Orr S, Itzkovitz S, Kashtan N, Chklovskii D, Alon U (2002) Network motifs: simple building blocks of complex networks. Science 298: 824-827.

[37] von Dassow G, Meir E, Munro EM, Odell GM (2000) The segment polarity network is a robust developmental module. Nature 406: 188-192.

[38] Davidson EH (2001) Genomic regulatory systems. Development and evolution. New York: Academic Press.

[39] Kauffman SA (1993) The Origins of Order: Self-Organization and Selection in Evolution. Oxford University Press: Oxford.

[40] de la Fuente A, Mendes P (2002) Quantifying gene networks with regulatory strengths. Mol Biol Rep 29: 73-77.

[41] Gama-Castro S, Jiménez-Jacinto V, Peralta-Gil M, Santos-Zavaleta A, Peñaloza-Spinola MI, et al. (2008) RegulonDB (version 6.0): gene regulation model of *Escherichia coli* K-12 beyond transcription, active (experimental) annotated promoters and Textpresso navigation. Nucleic Acids Res 36: D120-D124.

[42] Broder AZ, Kumar R, Maghoul F, Raghavan P, Rajagopalan S, Stata R, Tomkins A, Wiener, JL. (2000) Graph structure in the Web. Computer Networks 33: 309-320.

[43] Hong EL, Balakrishnan R, Dong Q, Christie KR, Park J, et al. (2008) Gene Ontology annotations at SGD: new data sources and annotation methods. Nucleic Acids Res 36: D577-D581.

[44] Solé RV, Valverde S (2008) Spontaneous emergence of modularity in cellular networks. J R Soc Interface 5: 129-133.

[45] Solé RV, Valverde S (2006) Are network motifs the spandrels of cellular complexity? Trends Ecol Evol 21: 419-422.

[46] Middendorf M, Ziv E, Wiggins CH (2005) Inferring network mechanisms: the *Drosophila melanogaster* protein interaction network. Proc Natl Acad Sci U S A 102: 3192-3197.

[47] Balaji S, Iyer LM, Aravind L, Babu MM. Uncovering a hidden distributed architecture behind scale-free transcriptional regulatory networks. J Mol Biol 360:204-212.

[48] Ash R *Information Theory*. Dover, New York, 1990.

[49] Barabási AL, Oltvai ZN. Network biology: understanding the cell's functional organization. Nat Rev Genet 5:101-113.

[50] Keseler IM, Collado-Vides J, Gama-Castro S,Ingraham J, Paley S, Paulsen IT, Peralta-Gil M, Karp PD. Ecocyc: a comprehensive database resource for escherichia coli. Nucleic Acids Res 33(Database issue):D334–D337.

[51] Newman MEJ. Assortative mixing in networks. Phys Rev Lett 89:208701.

[52] The particular definition of $\mathcal{W}$ depends on the specific approach used for dynamical approximation. In Boolean networks the effect of input combination, is explicitly defined in a truth table. Conversely, for neural, threshold approximations the activatory/inhibitory effect of TFs is commonly featured by a positive/negative value -weight- associated to every edge. Here $\mathcal{W}$ is a threshold function integrating the weight and the particular state at time t. In this work, we keep the general formalisation of $\mathcal{W}$ as a set of instructions relating combinatorial inputs and possible states

[53] Recursive pruning is a iterative deletion process used in other subgraph definitions as it is the case of k-core. However, our approach differs since it is a free parameter restricted for directed graphs.

[54] The concept of Dynamical Module is close to the topological interpretation of *essential set* in Markov Chains Theory. Consider a finite Markov Chain with states $s_1, ..., s_r$. A set of states B is said to be *essential* if every state in B is reachable (possibly in more than one step) from every other state in B and it is not possible to reach a state outside of B from a state in B [48]. Notice that the last condition is not required in our definition of DM.

[55] For the Erdös-Rènyi (ER) graph model $\langle C\rangle = \langle k\rangle/N$ and $\ell = \log N/\log\langle k\rangle$. Small world criteria implies $\langle C\rangle_R \gg \langle C\rangle_{ER}$ and $\ell_R \simeq \ell_{ER}$ (R subindex denotes the real network to be analyzed). In the case of ultra-small world condition, in addition to clustering criteria, the analysed network must fulfill $\ell \simeq \log(\log N)/\log\langle k\rangle$.