The Meaning of Structure in Interconnected Dynamic Systems

E. Yeung†, J. Gonçalves*, H. Sandberg◦, S. Warnick‡

Abstract—Interconnected dynamic systems are a pervasive component of our modern infrastructures. The complexity of such systems can be staggering, which motivates simplified representations for their manipulation and analysis. This work introduces the complete computational structure of a system as a common baseline for comparing different simplified representations. Linear systems are then used as a vehicle for comparing and contrasting distinct partial structure representations. Such representations simplify the description of a system’s complete computational structure at various levels of fidelity while retaining a full description of the system’s input-output dynamic behavior. Relationships between these various partial structure representations are detailed, and the landscape of new realization, minimality, and model reduction problems introduced by these representations is briefly surveyed.

I. INTRODUCTION

Structure and dynamic behavior are two of the most fundamental concepts characterizing a system. The interplay between these concepts has been a central theme in control as far back as Black’s, Bode’s, or Nyquist’s work on feedback amplifiers [1], [2], [3], or even as early as Maxwell’s analysis On Governors in 1868 [4].

The key property driving such analyses is the fact that an interconnection of systems yields another system. This property suggests a natural notion of structure, as the interconnection of systems, and focuses attention on understanding how interconnections of different systems result in varieties of dynamic behaviors.

This idea of structure as interconnection is not only critical in the analysis of systems, but it also plays a key role in system modeling. While black box approaches to modeling seek to duplicate the input-output behavior of a system irrespective of structure, first principles approaches to modeling use the visible interconnection structure of a system to decompose the system into smaller, fundamental components. Constitutive relations describing each of these components are then applied, and interconnection principles linking these relations then result in a model structurally and dynamically consistent with the original system.

While such approaches have demonstrated remarkable success in electrical and mechanical domains where system components are readily visible and physically separated from each other [5], [6], [7], application of such methods to biological, social, and other domains has been more difficult. One reason may be that these systems do not exhibit a natural structure in the same sense that previous applications have; while components of electrical and mechanical systems are compartmentalized and solid-state, the physical relationship among components of these other systems are often much more fluid [8], [9]. Perhaps for these other domains different notions of structure play the role historically occupied by the interconnection of components.

This paper explores these ideas by characterizing the complete computational structure of a system and then contrasting it with three distinct partial structure representations. These different representations include the interconnection of subsystems and the standard idea of a transfer function matrix, but it also includes a newer concept of system structure called signal structure that appears to be especially useful for characterizing systems that are difficult to compartmentalize. Precise relationships between these various perspectives of system structure are then provided, along with a brief discussion on their implications for various questions about realization and approximation.

II. COMPLETE COMPUTATIONAL STRUCTURE

The complete computational structure of a system characterizes the actual processes it uses to sense properties of its environment, represent and store variables internally, and affect change externally. At the core of these processes are information retrieval issues such as the encoding, storage, and decoding of quantities that drive the system’s dynamics. Different mechanisms for handling these quantities result in different system structures.

Mathematically, state equations, or their generalization as descriptor systems [10], [11], [12] are typically used to describe these mechanisms. Although there may be many realizations that describe the same input-output properties of a particular system, its complete computational structure is the architecture of the particular realization fundamentally used to store state variables in memory and transform system inputs to the corresponding outputs. In this work we will focus our attention on a class of differential algebraic systems that are equivalent to a set of ordinary differential equations in state space form; we will refer to such equations as generalized state equations.

Representing a system’s complete computational structure is thus a question of graphically representing the structure implied by the equations that govern its state evolution. In
this work, rather than focusing on the specific syntax of any one particular graphical modeling language, we will draw from the standard system theoretic notions of a block diagram and a signal flow graph to conduct a concrete analysis between graphical representations of a system at various levels of abstraction. The complete computational structure of a system, then, is the description of the system with the most refined resolution, which we will characterize as a graph derived from a particular block diagram of the generalized state equations.

To make this concept of structure precise, we begin by considering a system $G$ with generalized state space realization

$$
\begin{align*}
\dot{x} &= f(x,w,u), \\
w &= g(x,w,u), \\
y &= h(x,w,u).
\end{align*}
$$

(1)

Note that this system is in the form of a differential algebraic equation, although we will only consider systems with differentiation index zero, implying that (1) is always equivalent to a standard ordinary differential or difference equation of the same order [13]. Typically we may consider the system (1) to be defined over continuous time, with $r \in \mathbb{R} \geq 0$, and with $u \in \mathbb{R}^n$, $x \in \mathbb{R}^n$, $w \in \mathbb{R}^l$, $y \in \mathbb{R}^p$, and $\dot{x}$ taken to mean $dx/dt$. Moreover, we restrict our attention to those functions $f$, $g$ and $h$ where solutions exist for $t \geq 0$. Nevertheless, we could also consider discrete time systems, with $t = 0, 1, 2, 3, \ldots$ and $\dot{x}$ in (1) taken to mean $x[t+1]$, or systems with general input, state, auxiliary, and output spaces $\mathcal{U}$, $\mathcal{X}$, $\mathcal{W}$, or $\mathcal{Y}$, respectively. In some situations these “spaces” may merely be sets, e.g. $\mathcal{X} = \{0,1\}$. In any case, however, we will take $u \in \mathcal{U}^m$, $x \in \mathcal{X}^n$, $w \in \mathcal{W}^l$, and $y \in \mathcal{Y}^p$ so that $m, n, l$ and $p$ characterize the dimensions of the input, state, auxiliary and output vectors, respectively.

Note that the auxiliary variables, $w$, are used to characterize intermediate computation in the composition of functions. Thus, for example, we distinguish between $f(x) = x$ and $f(x) = 2(5x)$ by computing the latter as $f(w) = 2w$ and $w = g(x) = 5x$. In this way, the auxiliary variables serve to identify stages in the computation of the state space realization (1). Frequently we may not require any auxiliary variables in our description of the system; indeed it is the standard practice [14] to eliminate auxiliary variables to simplify the state descriptions of systems. Nevertheless, as we discuss structure, it will be critical to use auxiliary variables to distinguish between systems with dynamically equivalent, yet structurally distinct architectures, leading to the following definition.

**Definition 1:** Given a system (1), we call the number of auxiliary variables, $l$, the intricacy of the realization.

To understand the structure of (1), we need a notion of dependence of a function on its arguments. For example, the function $f(x,y,z) = xy - x + z$ clearly depends on $z$, but it only depends on $x$ when $y \neq 1$ (or on $y$ when $x \neq 0$). Since “structure” refers at some level to the dependence of the system variables on each other, it is important that our notion of dependence be made clear.

**Definition 2:** A function $f(w)$, from $l$-dimensional domain $\mathcal{W}$ to $s$-dimensional co-domain $\mathcal{X}$, is said to depend on the $j^{th}$ variable, $w_j$, if there exist values of the other $s-1$ variables $w_i, j \neq i$, such that $f(w)$ is not constant over all values of $w_i$ while holding the others variables fixed. If $s = 1$, then $f(w)$ depends on $w$ if it is not constant over all values of $w$.

Note that when $\partial f/\partial w_1$ is well defined, the above definition coincides with the partial derivative being non-zero for some value of the variables $w_j$. Nevertheless, here we allow for non-differentiable functions as we explicitly characterize one notion of the structure of a state space realization.

**Definition 3:** Given a system $G$ with realization (1), its complete or computational structure is a weighted directed graph $\mathcal{G}$ with vertex set $V(\mathcal{G})$, and edge set $E(\mathcal{G})$. The vertex set contains $m+n+l+p$ elements, with one for each input, state, auxiliary, and output variable of the system; we label the vertices accordingly. In particular, the vertex associated with the $j^{th}$ input is labeled $u_i$, $1 \leq i \leq m$, the vertex associated with the $j^{th}$ state is labeled $f_j$, $0 \leq j \leq n$, the vertex associated with the $k^{th}$ auxiliary variable is labeled $g_k$, $0 \leq k \leq l$, and the vertex associated with the $r^{th}$ output is labeled $h_r$, $1 \leq r \leq p$. The edge set contains an edge from node $i$ to node $j$ if the function associated with the label of node $j$ depends on the variable produced by node $i$. Moreover, the edge $(i,j)$ is then labeled (weighted) with the variable produced by node $i$.

So, for example, consider the following continuous time system with real-valued variables:

$$
\begin{align*}
\dot{x}_1 &= f_1(x_1, w_1), \\
\dot{x}_2 &= f_2(x_1), \\
w &= g(x_2), \\
y &= h(x_2).
\end{align*}
$$

(2)

Its complete computational structure has node set $V(\mathcal{G}) = \{u, f_1, f_2, g, h\}$ and edge set $E(\mathcal{G}) = \{u(f_1), x_1(f_1), x_1(f_2), x_2(f_2), y(1, f_1)\}$; Figure 1 illustrates the graph.

This notion of the complete computational structure of the system (1) corresponds with the traditional idea of the structure of a dynamic system (see for example [15], [16]), but with some important differences. First, this description uses auxiliary variables to keep track of the structural differences introduced by the composition of functions. This allows a degree of flexibility in how refined a view of the computational structure one considers “complete.” Also, this description may have some slight differences in the labeling of nodes and edges compared with various descriptions in the literature [15], [17], [18], [19], [14]. These differences will become important as new notions of partial structure are introduced in later sections. Here, we use rectangular nodes for graphs where the nodes represent systems, and the associated edges will represent signals. This convention will bridge well between the graphical representation of system structure and the typical engineering block diagram of a system, and it sometimes motivates the simplification in drawing edges as shown in Figure 1(b)), since every edge that leaves a node represents the same variable and carries the same label. Moreover, notice that nodes associated with the mechanisms that produce output variables are terminal, in that no edge ever leaves these nodes,
while the nodes associated with input variables are *sources*, in that no edge ever arrives at these nodes. Although it is common for engineering diagrams to explicitly draw the edges associated with output variables and leave them “dangling,” with no explicit terminal node, or to eliminate the input nodes and simply depict the input edges—also “dangling,” our convention ensures that the diagram corresponds to a well defined *graph*, with every edge characterized by an ordered pair of nodes. Note also that state nodes, such as $f_1$ in Figure 1, may have self loops, although auxiliary nodes will not, and at times it will be convenient to partition the vertex set into groups corresponding to the input, state, auxiliary, and output mechanisms as $V(\mathcal{C}) = \{V_u(\mathcal{C}), V_i(\mathcal{C}), V_w(\mathcal{C}), V_y(\mathcal{C})\}$. Likewise, we may similarly partition the edge set as necessary.

We see, then, that knowing the complete structure $\mathcal{C}$ of a system is equivalent to knowing its state space realization, along with the composition structure with which these functions are represented, given by $(f, g, h)$. We refer to this structure as *computational* because it reveals the dependencies among variables in the particular representation, or basis, that they are stored in and retrieved from memory. These specific, physical mechanisms that store and retrieve information, identified with $V_u(\mathcal{C})$, are interconnected with devices that transform variables, identified with $V_w(\mathcal{C})$, and with devices that interface with the system’s external environment. These devices include sensors, identified with $V_i(\mathcal{C})$, and actuators, identified with $V_y(\mathcal{C})$, to implement the particular system behavior observed by the outside world through the *manifest* variables, $u$ and $y$. Although other technologies very well may implement the same observed behavior via a different computational structure and a different representation of the *hidden* variables, $x$ and $w$, $\mathcal{C}$ describes the structure of the actual system employing existing technologies as captured through a particular state description. In this sense, $\mathcal{C}$ is the complete architecture of the system, and often may be interpreted as the system’s “physical layer.” Importantly, it is often this notion of structure, or a very related concept, that is meant when discussing the “structure” of a system, as the next example illustrates.

### A. Example: Graph Dynamical Systems

As an example, we examine the computational structure of a graph dynamical system (GDS). Graph dynamical systems are finite alphabet, discrete time systems with dynamics defined in terms of the structure of an associated undirected graph. They have been employed in the study of various complex systems [20], [21], including

- **Dynamical process on networks:**
  - disease propagation over a social contact graph,
  - packet flow in cell phone communication,
  - urban traffic and transportation;
- **Computational algorithms:**
  - Gauss-Seidel,
  - gene annotation based on functional linkage networks,
  - transport computations on irregular grids;
- **Computational paradigms related to distributed computing.**

Here we observe that the computational structure of the graph dynamical system corresponds naturally with the system’s underlying graph.

Given an undirected graph $\mathcal{G}$ with vertex set $V(\mathcal{G}) = \{1, 2, ..., n\}$, a GDS associates with each node $i$ a state $x_i$ that takes its values from a specified finite set $\mathcal{X}$. This state is then assigned a particular update function that updates its value according to the values of states associated with nodes adjacent to node $i$ on $\mathcal{G}$. Notice that this restriction on the update function suggests that the update function for state $i$ *depends* on states consistent with the structure of $\mathcal{G}$, indicating that the system’s computational structure $\mathcal{C}$ should correspond to the adjacency structure of $\mathcal{G}$.

The distinction is made between a GDS that updates all of its states simultaneously, called a *parallel GDS*, and one that updates its states in a particular sequence, called a *sequential GDS*. The parallel GDS thus becomes an autonomous dynamical system, evolving its state according to its update function from some initial condition. The sequential GDS, on the other hand, can be viewed as a controlled system that receives a particular permutation of the node set $V(\mathcal{G})$ as input and then evolves its states accordingly.

To illustrate, consider the sequential GDS given by $\mathcal{G} = \{1, 2, 3, 4\}$ as indicated in Figure 2. Let $\mathcal{S} = \{1, 2, 3, 4\}$ and...
The computational structure of the system (3) follows immediately from the dependency among variables characterized by equation (4); Figure 2 illustrates $G$ for this system. Notice that the structure of $G$ is reflected in $C$, where the undirected edges in $G$ have been replaced by directed edges in both directions, and self-loops have appeared where applicable. Moreover, notice that $C$ considers the explicit influence of the input sequence on the update computation, and explicitly identifies the system output. In this way, we see that the computational structure is a reflection of the natural structure of the sequential GDS.

B. Computational Structure of Linear Systems

Linear systems represent an important special case of those described by (1). They arise naturally as the linearization of sufficiently smooth nonlinear dynamics near an equilibrium point or limit cycle, or as the fundamental dynamics of systems engineered to behave linearly under nominal operating conditions. In either case, knowing the structure of the relevant linear system is a critical first step to understanding that of the underlying nonlinear phenomena.

The general state description of a linear system is given by

$$
\begin{align*}
\dot{x} &= Ax + \bar{A}w + Bu, \\
w &= \bar{A}x + \bar{A}w + Bu, \\
y &= Cx + \bar{C}w + Du,
\end{align*}
$$

(5)

where $A \in \mathbb{R}^{n \times n}$, $\bar{A} \in \mathbb{R}^{l \times n}$, $\bar{A} \in \mathbb{R}^{l \times l}$, $B \in \mathbb{R}^{n \times m}$, $\bar{B} \in \mathbb{R}^{l \times m}$, $C \in \mathbb{R}^{p \times l}$, and $D \in \mathbb{R}^{p \times m}$. Note that $I - \bar{A}$ is necessarily invertible, ensuring that the differentiability index of the system is zero. Nevertheless, the matrices are otherwise free.

As in the nonlinear case, it should be apparent that the auxiliary variables are superfluous in terms of characterizing the dynamic behavior of the system; this idea is made precise in the following lemma. Nevertheless, the auxiliary variables make a very important difference in terms of characterizing the system’s complete computational structure, as illustrated by the subsequent example.

**Lemma 1:** For any system (5) with intricacy $l > 0$, there exists a unique minimal intricacy realization $(A_0, B_0, C_0, D_0)$ with $l = 0$ such that for every solution $(u(t), x(t), w(t), y(t))$ of (5), $(u(t), x(t), y(t))$ is a solution of $(A_0, B_0, C_0, D_0)$.

**Proof:** The result follows from the invertibility of $(I - \bar{A})$. Solving for $w$ and substituting into the equations of $x$ and $y$ then yields $(A_0, B_0, C_0, D_0)$.
Consider, for example, the system (5) with state matrices given by \( D = 0 \) and the following:

\[
A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \hat{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}
\]

\[
\bar{A} = \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix}, \quad \bar{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_1 & 0 \\ 0 & a_2 \end{bmatrix}
\]

\[
B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \bar{B}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & b_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & b_2 & 0 & 0 \end{bmatrix}
\]

\[
C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \bar{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]

This system has the complete computational structure \( \mathcal{C} \) shown in Figure 3(a). Here, because each auxiliary variable is defined as the simple product of a coefficient times another variable, we label the node corresponding to \( w_i \) in \( \mathcal{C} \) with the appropriate coefficient rather than the generic label, \( g_i \). Note that this realization has an intricacy of \( l = 8 \).

Suppose, however, that we eliminate the last six auxiliary variables, leading to an equivalent realization with intricacy \( l = 2 \). The state matrices then become

\[
A = \begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix}, \quad \hat{A} = \begin{bmatrix} 0 & e_1 \\ e_2 & 0 \end{bmatrix}
\]

\[
\bar{A} = \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix}, \quad \bar{A} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}
\]

\[
B = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}, \quad \bar{B}^T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}
\]

\[
C = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \bar{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

with computational structure \( \mathcal{C} \) as shown in Figure 3(b). Similarly, we can find an equivalent realization with \( l = 0 \) given by

\[
A_o = \begin{bmatrix} a_1 & e_1 c_2 \\ e_2 c_1 & a_2 \end{bmatrix}, \quad B_o = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}, \quad C_o = \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix}
\]

and all other system matrices equal to zero. This realization, (8), is the minimal intricacy realization of both systems (6) and (7), and its complete computational structure \( \mathcal{C} \) is given in Figure 3(c). The equivalence between these realizations is easily verified by substitution.

Comparing the computational structures for different realizations of the same system, (6), (7), and (8), we note that the intricacy of auxiliary variables plays a critical role in suppressing or revealing system structure. Moreover, note that auxiliary variables can change the nature of which variables are manifest or hidden. In Figure 3 shaded regions indicate which variables, represented by edges, are hidden; manifest variables leave shaded regions while hidden variables are contained within them. Note that the minimal intricacy realization...
has no internal manifest variables, or, in other words, it has a single block of hidden variables (Figure 3(c)). Meanwhile, both \( w_1 \) and \( w_2 \) are manifest in the other realizations (Figures 3(a), 3(b)) since \( w_1 = y_1 \) and \( w_2 = y_2 \), indicated by the “1” at their respective terminal nodes. This yields two distinct blocks of hidden variables, in either case, revealing the role intricacy of a realization can play characterizing its structure.

The complete computational structure of a system is thus a graphical representation of the dependency among input, state, auxiliary, and output variables that is in direct, one-to-one correspondence with the system’s state realization, generalized to explicitly account for composition intricacy. All structural and behavioral information is fully represented by this description of a system. Nevertheless, this representation of the system can also be unwieldy for large systems with intricate structure.

III. PARTIAL STRUCTURE REPRESENTATIONS

Complex systems are often characterized by intricate computational structure and complicated dynamic behavior. State descriptions and their corresponding complete computational structures accurately capture both the system’s structural and dynamic complexity, nevertheless these descriptions themselves can be too complicated to convey an efficient understanding of the nature of the system. Simplified representations are then desirable.

One way to simplify the representation of a system is to restrict the structural information of the representation while maintaining a complete description of the system’s dynamics. The most extreme example of this type of simplified representation is the transfer function of a single-input single-output linear time invariant (LTI) system. A transfer function completely specifies the system’s input-output dynamics without retaining any information about the computational structure of the system. For example, consider the \( n^{th} \) order LTI single-input single-output system given by \( (A, b, c, d) \). It is well known that although the state description of the system completely specifies the transfer function, \( G(s) = c(sI - A)^{-1}b + d \), the transfer function \( G(s) \) has an infinite variety of state realizations, and hence computational structures, that all characterize the same input-output behavior. That is, the structural information in any state realization of the system is completely removed in the transfer function representation of the system, even though the dynamic (or behavioral) information about the system is preserved.

We use this power of a transfer function to obfuscate structural information to develop three distinct partial-structure representations of an LTI system: subsystem structure, signal structure, and the sparsity pattern of a (multiple input, multiple output) system’s transfer function matrix. Later we will show how each of these representations contain different kinds of structural information, and we will precisely characterize the relationships among them.

A. Subsystem Structure

One of the most natural ways to reduce the structural information in a system’s representation is to partition the nodes of its computational structure into subsystems, then replace these subsystems with their associated transfer function. Each transfer function obfuscates the structure of its associated subsystem, and the remaining (partial) structural information in the system is the interconnection between transfer functions.

Subsystem structure refers to the appropriate decomposition of a system into constituent subsystems and the interconnection structure between these subsystems. Abstractly, it is the condensation graph of the complete computational structure graph, \( E \), taken with respect to a particular partition of \( E \) that identifies subsystems in the system. Such abstractions have been used in various ways to simplify the structural descriptions of complex systems [15], [22], for example by “condensing” strongly connected components or other groups of vertices of a graph into single nodes. Nevertheless, in this work we define a particular condensation graph as the subsystem structure of the system. We begin by characterizing the partitions of \( E \) that identify subsystems.

Definition 4: Given a system \( G \) with realization (5) and associated computational structure \( E \), we say a partition of \( V(E) \) is admissible if every edge in \( E(V(E)) \) between components of the partition represents a variable that is manifest, not hidden.

For example, considering the system (8) with \( V(E) = \{u_1, f_1, c_1, c_2, f_2, u_2\} \). We see that the partition \( \{\{u_1\}, \{f_1, c_1, c_2, f_2\}, \{u_2\}\} \) is admissible since the only edges between components are \( u_1(1, f_1) \) and \( u_2(2, f_2) \), representing the manifest variables \( u_1 \) and \( u_2 \). Notice that the shading in Figure 3(c) is consistent with this admissible partition. Alternatively, the partition \( \{\{u_1\}, \{f_1, c_1\}, \{c_2, f_2\}, \{u_2\}\} \) is not admissible for (8), since the edges \( x_1(f_1, f_2) \) and \( x_2(f_2, f_1) \) extend between components of the partition but represent variables \( x_1 \) and \( x_2 \) that are hidden, not manifest.

Although sometimes any aggregation, or set of fundamental computational mechanisms represented by vertices in \( E \), may be considered a valid subsystem, in this work a subsystem has a specific meaning. In particular, the variables that interconnect subsystems must be manifest, and thus subsystems are identified by the components of admissible partitions of \( V(E) \). We adopt this convention to 1) enable the distinction between real subsystems vs. merely arbitrary aggregations of the components of a system, and 2) ensure that the actual subsystem architecture of a particular system is adequately reflected in the system’s computational structure and associated realization, thereby ensuring that such realization is complete.

Definition 5: Given a system \( G \) with realization (5) and associated computational structure \( E \), the system’s subsystem structure is a condensation graph \( V(\mathcal{F}) \) of \( E \) with vertex set \( V(\mathcal{F}) \) and edge set \( E(\mathcal{F}) \) given by:

- \( V(\mathcal{F}) = \{S_1, ..., S_8\} \) are the elements of an admissible partition of \( V(E) \) of maximal cardinality, and
- \( E(\mathcal{F}) \) has an edge \( (S_i, S_j) \) if \( E(\mathcal{F}) \) has an edge from some component of \( S_i \) to some component of \( S_j \).
We label the nodes of $V(\mathcal{G})$ with the transfer function of the associated subsystem, which we also denote $S_i$, and the edges of $E(\mathcal{G})$ with the associated variable from $E(\mathcal{G})$.

Note that, like $\mathcal{G}$, the subsystem structure $\mathcal{F}$ is a graph with vertices that represent systems and edges that represent signals, or system variables. For example, Figure 4(a) illustrates the subsystem structure for both systems (6) and (7), shown in Figures 3(a) and 3(b). Note that the subsystem structure of these systems’ minimally intricate realization, (8), is quite different, with a single block rather than two blocks interconnected in feedback, as shown in Figure 4(b). This illustrates the necessity of auxiliary variables to adequately describe the complete system structure.

**Fig. 4.** The subsystem structure of a linear system partitions vertices of the complete computational structure and condenses admissible groups of nodes into single subsystem nodes, shown in brown; nodes shown in green are those that correspond directly to unaggregated vertices of the complete computational structure, which will always be associated with mechanisms that generate manifest variables. Here, the subsystem structure corresponds to the interconnection of shaded regions in Figure 3. Comparing Figures 4(a) and 4(b), we note that intricacy in a realization may be necessary to characterize meaningful subsystems and yield nontrivial subsystem structure.

**Lemma 2:** The subsystem structure $\mathcal{F}$ of a system $G$, with complete computational structure $\mathcal{G}$, is unique.

**Proof:** We prove by contradiction. Suppose the subsystem structure $\mathcal{F}$ of $G$ is not unique. Then there are at least two distinct subsystem structures of $G$, which we will label $\mathcal{F}^1$ and $\mathcal{F}^2$. This implies there are two admissible partitions of $V(\mathcal{G})$, given by $V(\mathcal{F}^1)$ and $V(\mathcal{F}^2)$, such that $V(\mathcal{F}^1) \neq V(\mathcal{F}^2)$ and with equal cardinality, $q$. Note that by definition, $q$ must be the maximal cardinality of any admissible partition of $V(\mathcal{G})$. To obtain a contradiction, we will construct another admissible partition, $V(\mathcal{F}^3)$, such that $|V(\mathcal{F}^3)| > q$.

Consider the following partition of $V(\mathcal{G})$ that is a refinement of both $V(\mathcal{F}^1)$ and $V(\mathcal{F}^2)$:

$$V(\mathcal{F}^3) = \{S^3|S^3 \neq \emptyset; S^3 = S_i \cap S_j, S_i \in V(\mathcal{F}^1), S_j \in V(\mathcal{F}^2)\}.$$  

Since $V(\mathcal{F}^1) \neq V(\mathcal{F}^2)$, then $|V(\mathcal{F}^3)| > q$, since the cardinality of the refinement must be greater than that of $V(\mathcal{F}^1)$ or $V(\mathcal{F}^2)$. Moreover, note that the partition $V(\mathcal{F}^3)$ is admissible, since every edge of $\mathcal{G}$ between vertices associated with distinct components of $V(\mathcal{F}^3)$ corresponds with an edge of either $\mathcal{F}^1$ or $\mathcal{F}^2$, which are admissible. Thus, $V(\mathcal{F}^3)$ is an admissible partition of $V(\mathcal{G})$ with cardinality greater than $q$, which contradicts the assumption that $\mathcal{F}^1$ and $\mathcal{F}^2$ are both subsystem structures of $G$.

- The subsystem structure of a system reveals the way natural subsystems are interconnected, and it can be represented in other ways besides (but equivalent to) specifying $\mathcal{F}$. For example, one common way to identify this kind of subsystem architecture is to write the system as the linear fractional transformation (LFT) with a block diagonal “subsystem” component and a static “interconnection” component (see [23] for background on the LFT). For example, the system in Figure 4(a) can be equivalently represented by the feedback interconnection of a static system $N: U \times W \to V \times (U \times W)$ and a block-diagonal dynamic system $S: U \times W \to W$ given by

$$N = \begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0
\end{bmatrix}, \quad S = \begin{bmatrix}
S_1 & 0 \\
0 & S_2
\end{bmatrix},$$

where

$$S_1: \begin{bmatrix} u_1 \\ w_2 \end{bmatrix} \to w_1 \quad S_2: \begin{bmatrix} w_1 \\ u_2 \end{bmatrix} \to w_2$$

In general, the LFT associated with $\mathcal{F}$ will have the form

$$N = \begin{bmatrix}
0 & I & L & K
\end{bmatrix} \quad S = \begin{bmatrix}
S_1 & 0 & \ldots & 0 \\
0 & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots \\
0 & \ldots & 0 & S_q
\end{bmatrix},$$

where $q$ is the number of distinct subsystems, and $L$ and $K$ are each binary matrices of the appropriate dimension (see Figure 5). Note that if additional output variables are present, besides the manifest variables used to interconnect subsystems, then the structure of $N$ and $S$ above extend naturally. In any event, however, $N$ is static and $L$ and $K$ are binary matrices.

The subsystem structure of any system is well defined although it may be trivial (a single internal block) if the system does not decompose naturally into an interconnection of subsystems, such as (8) in Figure 3(c) and Figure 4(b). Note that $\mathcal{F}$ always identifies the most refined subsystem structure possible, and for systems with many interconnected subsystems, coarser representations may be obtained by aggregating various subsystems together and constructing the resulting condensation graph. These coarser representations effectively absorb some interconnection variables and their...
associated edges into the aggregated components, suggesting that such representations are the subsystem structure for less intricate realizations of the system, where some of the manifest auxiliary variables are removed, or at least left hidden. The subsystem structure is thus a natural partial description of system structure when the system can be decomposed into the interconnection of specific subsystems.

**B. Signal Structure**

Another very natural way to partially describe the structure of a system is to characterize the direct causal dependence among each of its manifest variables; we refer to this notion as the *signal structure*. This description of the structure of a system makes no attempt to cluster, or partition, the actual internal system states. As a result, it offers no information about the internal interconnection of subsystems, and signal structure can therefore be a very different description of system structure than subsystem structure.

Given a generalized linear system (5) with complete computational structure \( \mathcal{C} \), we characterize its signal structure by considering its minimal intricacy realization \( (A_o, B_o, C_o, D_o) \). We assume without loss of generality that the outputs \( y \) are ordered in such a way that \( C_o \) can be partitioned

\[
C_o = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}
\]

where \( C_{11} \in \mathbb{R}^{p_1 \times p_1} \) is invertible, with \( p_1 \) equal to the rank of \( C_{11} \); \( C_{12} \in \mathbb{R}^{p_1 \times (n-p_1)} \); \( C_{21} \in \mathbb{R}^{(p-p_1) \times (n-p_1)} \); and \( C_{22} \in \mathbb{R}^{(p-p_1) \times p_1} \). Note that if the outputs of the minimal intricacy realization do not result in \( C_{11} \) being invertible, then it is possible to reorder them so the first \( p_1 \) outputs correspond to independent rows of \( C_o \); the states can then be renumbered so that \( C_{11} \) is invertible. One can show that such reordering of the outputs and states of the minimal intricacy realization only affects the ordering of the states and outputs of the original system; the graphical relationship of the computational structure is preserved.

The direct causal dependence among manifest variables is then revealed as follows. First, since \( C_o \) is rank \( p_1 \), the rank-nullity theorem guarantees \( C_o \) has nullity \( n-p_1 \). Let

\[
N_{n-p_1} = \begin{bmatrix} N_1 \\ N_2 \end{bmatrix},
\]

where \( N_1 \in \mathbb{R}^{p_1 \times (n-p_1)} \), \( N_2 \in \mathbb{R}^{(n-p_1) \times (n-p_1)} \), be a matrix of column vectors that form a basis for the null space of \( C_o \). By the definition of \( N \) and the invertibility of \( C_{11} \), we can write

\[
N = \begin{bmatrix} -C_{11}^{-1}C_{12}N_2 \\ N_2 \end{bmatrix}.
\]

Since the columns of \( N \) form a basis, we deduce that \( N_2 \) is invertible; thus the state transformation \( z = Tx \) given by

\[
T = \begin{bmatrix} C_{11} & C_{12} \\ 0 & N_2^{-1} \end{bmatrix},
\]

is well defined. This transformation yields a system of the form

\[
\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u
\]

\[
\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} I & 0 \\ C_2 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} u
\]

where \( C_2 = C_{21}C_{11}^{-1} \), \( z_1 \in \mathbb{R}^{p_1} \), \( z_2 \in \mathbb{R}^{n-p_1} \), \( u \in \mathbb{R}^m \), \( y_1 \in \mathbb{R}^{p_1} \), and \( y_2 \in \mathbb{R}^{n-p_1} \). To simplify the expression we will abuse notation and refer to the above system as \( (A, B, C, D) \), since there is little opportunity to confuse these matrices with those of the original system given in (5). In fact, the system (13) is simply a change of coordinates of the minimal intricacy realization \( (A_o, B_o, C_o, D_o) \), possibly with a reordering of the output and state variables. The direct causal dependence among manifest variables is then revealed by the dynamical structure function of \( (A, B, \begin{bmatrix} I & 0 \end{bmatrix}, D) \).

The dynamical structure function class of linear systems was defined in [24] and discussed in [25], [26], [27], [28], [29], [30]. This representation of a linear system describes the direct causal dependence among a subset of state variables, and it will extend to characterize signal structure for the system in (13). We repeat and extend the derivation here to demonstrate its applicability to the system (13). Taking Laplace transforms and assuming zero initial conditions yields the following relationships

\[
\begin{bmatrix} sZ_1 \\ sZ_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} U
\]

where \( Z(s) \) denotes the Laplace transform of \( z(t) \), etc. Solving for \( Z_2 \) in the second equation and substituting into the first then yields

\[
sZ_1 = W(s)Z_1 + V(s)U
\]
where $W(s) = [A_{11} + A_{12}(sI - A_{22})^{-1}A_{21}]$ and $V(s) = [B_1 + A_{12}(sI - A_{22})^{-1}B_2]$. Let $\hat{D}(s)$ be the matrix of the diagonal entries of $W(s)$, yielding

$$Z_1 = Q(s)Z_1 + P(s)U$$

(16)

where $Q(s) = (sI - \hat{D}(s))^{-1}(W(s) - \hat{D}(s))$ and $P(s) = (sI - \hat{D}(s))^{-1}V(s)$. From (13) we note that $Z_1 = Y_1 - D_1U$, which, substituting into (16), then yields:

$$\begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} Q(s) \\ C_2 \end{bmatrix} Y_1 + \begin{bmatrix} P(s) + (I - Q(s))D_1 \\ D_2 \end{bmatrix} U \quad (17)$$

We refer to the matrices

$$\begin{bmatrix} Q(s) & C_2 \\ C_2^T & 0 \end{bmatrix}^{T} \begin{bmatrix} P(s) + (I - Q(s))D_1 & D_2 \end{bmatrix}^{T}$$

as $Q$ and $P$, respectively. The matrices $(Q(s), P(s))$ are called the dynamical structure function of the system (13), and they characterize the dependency graph among manifest variables as indicated in Equation (17). We note a few characteristics of $(Q(s), P(s))$ that give them the interpretation of system structure, namely:

- $Q(s)$ is a square matrix of strictly proper real rational functions of the Laplace variable, $s$, with zeros on the diagonal. Thus, if each entry of $y_1$ were the node of a graph, $Q_{ij}(s)$ would represent the weight of a directed edge from node $j$ to node $i$; the fact $Q_{ij}(s)$ is proper preserves the meaning of the directed edge as a causal dependency of $y_1$ on $y_j$.
- Similarly, the entries of the matrix $[P(s) + (I - Q(s))D_1]$ are proper and thus carry the interpretation of causal weights characterizing the dependency of entries of $y_1$ on the $m$ inputs, $u$. Note that when $D_1 = 0$, this matrix reduces to $P(s)$, which has strictly proper entries.

This leads naturally to the definition of signal structure.

**Definition 6:** The **signal structure** of a system $G$, with realization (5) and equivalent realization (13), and with dynamical structure function $(Q(s), P(s))$ characterized by (16), is a directed graph $\mathcal{W}$, with a vertex set $V(\mathcal{W})$ and edge set $E(\mathcal{W})$ given by:

- $V(\mathcal{W}) = \{ u_1, ..., u_m, y_{11}, ..., y_{1n}, y_{21}, ..., y_{2m} \}$, each representing a manifest signal of the system, and
- $E(\mathcal{W})$ has an edge from $u_i$ to $y_{1j}$, $u_i$ to $y_{2j}$, $y_{1j}$ to $y_{1l}$ or $y_{1l}$ to $y_{2j}$, if the associated entry in $[P(s) + (I - Q(s))D_1]$, $D_2$, $Q(s)$, or $C_2$ (as given in Equations (16) and (17)) is nonzero, respectively.

We label the nodes of $V(\mathcal{W})$ with the name of the associated variable, and the edges of $E(\mathcal{W})$ with the associated transfer function entry from Equation (17).

Signal structure is fundamentally a different type of graph than either the computational or subsystem structure of a system because, unlike these other graphs, vertices of a system’s signal structure represent *signals* rather than systems. Likewise, the edges of $\mathcal{W}$ represent *systems* instead of signals, as opposed to $\mathcal{C}$ or $\mathcal{S}$. We highlight these differences by using circular nodes in $\mathcal{W}$, in contrast to the square nodes in $\mathcal{C}$ or $\mathcal{S}$. The next example illustrates a system without notable subsystem structure and no apparent structural motif in its complete computational structure; nevertheless, it reveals a simple and elegant ring structure in its signal structure.

**Example 1:** Ring Signal Structure. Systems with no apparent structure in any other sense can nevertheless possess a very particular signal structure. Consider the minimally intricate linear system, specified by the state-space realization $(A_o, B_o, C_o, D_o)$, where

$$A_o = \frac{1}{12} \begin{bmatrix} -178 & 262 & -10 & -141 & -19 & 88 \\ -156 & 252 & -12 & -156 & -48 & 84 \\ -158 & 266 & -38 & -147 & -5 & 128 \\ -12 & 48 & -12 & -72 & -12 & 12 \\ -288 & 504 & 0 & -264 & -180 & 144 \\ 0 & 24 & 0 & -24 & -12 & -12 \end{bmatrix},$$

$$B_o = \frac{1}{4} \begin{bmatrix} 0 & -1 & 21 \\ 0 & 0 & 12 \\ -8 & 1 & 27 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$C_o = \begin{bmatrix} -1 & 4 & -1 & -2 & -1 & 1 \\ -12 & 21 & 0 & -11 & -5 & 6 \end{bmatrix}$$

$$D_o = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

(18)

![Fig. 6. The complete computational structure $\mathcal{C}$, of the system $(A_o, B_o, C_o, D_o)$ given by (18). Here, edge labels, $u$ and $x$, and self loops on each node $j$ have been omitted to avoid the resulting visual complexity. Edges associated with variables $x$, which are not manifest, are entirely contained within the shaded region (which also corresponds to the strong partial condensation shown in Figure 7(a)).](image-url)

We compute the signal structure by employing a change of coordinates on the state variables to find an equivalent...
The subsystem structure, \( S \), of the system shown in Figures 6 and 7(b); the system has no interconnection structure between subsystems because the system is composed of only a single subsystem; it does not meaningfully decompose into smaller subsystems.

The signal structure, \( W \), of the system shown in Figures 6 and 7(a). Note that, in contrast with \( S \) and \( \mathcal{S} \), vertices of \( W \) represent manifest signals (characterized by round nodes), while edges represent systems.

Fig. 7. Distinct notions of partial structure for the system \((A_p,B_p,C_p,D_p)\) given by Equation (18). Observe that while the system is unstructured in the strong sense, it is actually quite structured in the weak sense. Moreover, although the subsystem structure is visible in the complete computational structure (Figure 6) as a natural condensation (note the shaded region), the signal structure is not readily apparent.

realization of the form (13). The transformation

\[
A = T A_0 T^{-1} = \begin{bmatrix}
-2 & 0 & 0 & 0 & 0 & -3 \\
-3 & 0 & -1 & 0 & 0 & 0 \\
0 & -4 & 0 & 5 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 5 & 0 \\
0 & 0 & 1 & 0 & 0 & -1 \\
\end{bmatrix}
\]

\[
B = T B_0 = \begin{bmatrix}
2 & 0 & 0 \\
0 & 3 & 0 \\
0 & 0 & 6 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

\[C = C_0 T^{-1} = \begin{bmatrix}
I_3 & 0 \\
\end{bmatrix},
\]

\[D = D_0 = [0].\]

The dynamical structure function of the system, \((Q,P)\), then becomes

\[
Q = \begin{bmatrix}
0 & 0 & \frac{-3}{s^2 + 3s + 2} \\
\frac{-1}{s^2 + 3s + 2} & 0 & 0 \\
0 & \frac{10}{s^2 + 9s + 20} & 0 \\
\end{bmatrix},
\]

\[
P = \begin{bmatrix}
\frac{2}{s^2} & 0 & 0 \\
0 & \frac{3}{s^2} & 0 \\
0 & 0 & \frac{6}{s^2 + 4} \\
\end{bmatrix}
\]

which yields the signal structure, \( W \), as shown in Figure 7(b). Notice that although the complete computational structure and subsystem structure do not characterize any meaningful interconnection patterns, the system is nevertheless structured and organized in a very concrete sense. In particular, the outputs \( y_1, y_2, \) and \( y_3 \) form a cyclic dependency, and each causally depends on a single input \( u_i, i = 1,2,3, \) respectively.

C. Sparsity Structure of the Transfer Function Matrix

The weakest notion of structure exhibited by a system is the pattern of zeros portrayed in its transfer function matrix, where “zero” refers to the value of the particular transfer function element, not a transmission zero of the system. Like signal structure, this type of structure is particularly meaningful for multiple-input multiple-output systems, and, like signal structure, the corresponding graphical representation reflects the dependance of system output variables on system input variables. Thus, nodes of the graph will be signals, represented by circular nodes, and the edges of the graph will represent systems, labeled with the corresponding transfer function element; a zero element thus corresponds to the absence of an edge between the associated system input and output. Formally, we have the following definition.

**Definition 7:** The sparsity structure of a system \( G \) is a directed graph \( \mathcal{G} \), with a vertex set \( V(\mathcal{G}) \) and edge set \( E(\mathcal{G}) \) given by:

- \( V(\mathcal{G}) = \{ u_1,...,u_m,y_1,...,y_p \} \), each representing a manifest signal of the system, and
Such an effect, however, could be the result of certain internal cancellations and does not suggest, for example, that there is no path in the complete computational structure from the $j^{th}$ input to the $i^{th}$ output. Thus, for example, a diagonal transfer function matrix does not imply the system is decoupled.

The next example demonstrates that a fully coupled system may, nevertheless, have a diagonal transfer function, even when the system is minimal in both intricacy and order. That is, the internal cancellations necessary to generate the diagonal sparsity structure in the transfer function while being fully coupled do not result in a loss of controllability or observability. Thus, the sparsity structure only describes the input-output dependencies of the system and does not imply anything about the internal information flow or communication structure. This fact is especially important in the context of decentralized control, where the focus is often on shaping the sparsity of a controller’s transfer function given a particular sparsity in the transfer function of the system to be controlled.

Example 2: Diagonal Transfer Function $\neq$ Decoupled. Consider a system, $G$, with the following minimal intricacy realization, $(A_o, B_o, C_o, D_o)$:

$$
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
-5 & 1 \\
2 & -4
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
2 & 1 \\
4 & -1
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2
\end{bmatrix}
$$

(23)

$$
\begin{bmatrix}
y_1 \\
y_2
\end{bmatrix} =
\begin{bmatrix}
1 & 1 \\
-4 & 2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
$$

(24)

IV. RELATIONSHIPS AMONG REPRESENTATIONS OF STRUCTURE

In this section, we explore the relationships between the four representations of structure defined above. What we will find is that some representations of structure are more informative than others. Next, we will discuss a specific class of systems in which the four representations of structure can be ranked by information content (with one representation encapsulating all the information contained in another representation of structure). Outside this class of systems, however, we will demonstrate that signal structure and subsystem structure have fundamental differences in addition to those arising from the graphical conventions of circular versus square nodes, etc. Signal and subsystem structure provide alternative ways of discussing a system’s structure without requiring the full detail of a state-space realization or the abstraction imposed by a transfer function. Understanding the relationships between these representations then enables the study of new kinds of research problems that deal with the realization, reconstruction, and approximation of system structure (where structure can refer to the four representations defined so far or other...
representations of structure not mentioned in this work). The final section gives a discussion of such future research.

Different representations of structure contain different kinds of structural information about the system. For example, the complete computational structure details the structural dependencies among fundamental units of computation. Using complete computational structure to model system structure requires knowledge of the parameters associated with each unit of computation. Partial representations of structure such as signal and subsystem structure do not require knowledge of such details in their description. Specifically, subsystem structure essentially aggregates units of computation to form subsystems and models the closed-loop transfer function of each subsystem. Signal structure models the SISO transfer functions describing direct causal dependencies between outputs and inputs of some of the fundamental units of computation that happen to be manifest. Sparsity structure models the closed-loop dependencies of system outputs on inputs. Thus, complete computational structure appears to be the most demanding or information-rich description of system structure. This intuition is made precise with the following result:

**Theorem 1:** Suppose a complete computational structure has minimal intricacy realization \((A_o, B_o, C_o, D_o)\) with

\[
C_0 = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}
\]

and \(C_{11}\) invertible. Then the complete computational structure specifies a unique subsystem, signal, and sparsity structure.

**Proof:** Let \(\mathcal{C}\) be a computational structure with minimal intricacy realization \((A_o, B_o, C_o, D_o)\) with

\[
C_o = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},
\]

\(C_{11}\) invertible. By Lemma 2, the subsystem structure \(\mathcal{S}\) is unique. Since \(C_{11}\) is invertible, we see that equations (15) and (17) imply that the minimal intricacy realization uniquely specifies the dynamical structure function of the system. By definition, the signal structure is unique. Finally, write the generalized state-space realization of \(\mathcal{C}\) as

\[
(A, B, C, D) = \left( \begin{bmatrix} A & \hat{A} \\ \bar{A} & \bar{A} \end{bmatrix}, \begin{bmatrix} B \\ \bar{B} \end{bmatrix}, \begin{bmatrix} C & \bar{C} \end{bmatrix}, D \right).
\]

The uniqueness of the sparsity structure follows from its one-to-one correspondence with the transfer function \(G(s) = C_o(sI - A_o)^{-1}B_o + D_o\) which can also be expressed as

\[
(C + \bar{C}(I - \bar{A})^{-1}\hat{A})(sI - A - \bar{A}(I - \bar{A})^{-1}\hat{A})^{-1}(B + \bar{B}(I - \bar{A})^{-1}\bar{A}).
\]

Similarly, subsystem, signal, and sparsity structure can be realized using multiple complete computational structures. Without additional assumptions, it is impossible to associate a unique complete computational structure with a given subsystem, signal, or sparsity structure. Theorem 1 shows that a complete computational structure specifies a unique subsystem, signal, and sparsity structure. In this sense, a complete computational structure is a more informative description of system structure than subsystem, signal and sparsity structure. The next result has a similar flavor and follows directly from the one-to-one correspondence of a system’s transfer function with its sparsity structure.

**Theorem 2:** Every subsystem structure or signal structure specifies a unique sparsity structure.

**Proof:** Consider the LFT representation \(\mathcal{F}(N,S)\) of a subsystem structure \(\mathcal{S}\); write

\[
N = \begin{bmatrix} 0 & I \\ L & K \end{bmatrix}
\]

as in equation (11). The linear fractional transform gives the input-output map, i.e. the transfer function. Thus, \(G(s) = (I - SK)^{-1}SL\).

Similarly, using the dynamical structure representation of the signal structure \(\mathcal{S}\) given in equation (17), we can solve for the transfer function

\[
G(s) = \left(I - \begin{bmatrix} Q(s) & 0 \\ C_2 & 0 \end{bmatrix}\right)^{-1}\begin{bmatrix} P(s) + (I - Q(s))D_1 \\ D_2 \end{bmatrix}
\]

The result follows from the definition of sparsity structure.

The relationship between subsystem structure and signal structure is not so straightforward. Nevertheless, subsystem structure does specify a unique signal structure for a class of systems, namely systems with subsystem structure composed of single output (SO) subsystems and where every manifest variable is involved in subsystem interconnection. For this class of systems, subsystem structure is a richer description of system structure than signal structure.

**A. Single Output Subsystem Structure and Signal Structure**

**Theorem 3:** Let \(\mathcal{S}\) be a SO subsystem structure with LFT representation \(\mathcal{F}(N,S)\). Suppose

\[
N = \begin{bmatrix} 0 & I \\ L & K \end{bmatrix}
\]

where \([L \ K]\) has full column rank. Then \(\mathcal{S}\) uniquely specifies the signal structure of the system.

**Proof:** The dynamics of \(N\) and \(S\) satisfy

\[
\begin{align*}
Y &= [0]U + [I]Y \\
\pi &= LU + KY \\
Y &= S\pi
\end{align*}
\]

Combining the second and third equation, we get

\[
Y = S\pi = S\left[\begin{bmatrix} K \\ L \end{bmatrix} \begin{bmatrix} Y \\ U \end{bmatrix}\right].
\]
Since \( \mathcal{J} \) is a SO subsystem structure, the entries of \( S \) describe direct causal dependencies among manifest variables involved in interconnection. Since \( [L \ K] \) has full column rank and is a binary matrix, this means that each manifest variable is used at least once in interconnection. Thus, \( S \) describes the direct causal dependencies between all manifest variables of the system and specifies the signal structure of the system. Notice that for the class of systems described above, the four representations of structure can be ordered in terms of information content. Theorem 1 shows that the complete computational structure uniquely specifies all the other representations of structure and thus is the most informative of the four. By Theorem 3 and Theorem 2 respectively, subsystem structure uniquely specifies the signal structure and sparsity structure of the system and thus is the second most informative. Similarly, signal structure is the third most informative and sparsity structure is the least informative of the four representations of structure.

We note that the converse of Theorem is also true, namely if the subsystem structure of a system specifies a unique signal structure then the subsystem structure is a SO subsystem structure where every manifest variable is an interconnection variable. The proof is simple and follows from the result of the next subsection. We also provide several examples that show how a multiple output subsystem structure can be consistent with multiple signal structures - these all will serve to illustrate the general relationship between subsystem and signal structure outside of the class of systems mentioned above.

### B. The Relationship Between Subsystem and Signal Structure

Subsystem structure and signal structure are fundamentally different descriptions of system structure. In general, subsystem structure does not encapsulate the information contained in signal structure. Signal structure describes direct causal dependencies between manifest variables of the system. Subsystem structure describes closed loop dependencies between manifest variables involved in the interconnection of subsystems. Both representations reveal different perspectives of a system’s structure. The next result makes this relationship between subsystem and signal structure precise.

**Theorem 4:** Given a system \( G \), let \( \mathcal{F}(N,S) \) be the LFT representation of a subsystem structure \( \mathcal{J} \). In addition, let the signal structure of the system \( G \) be denoted as in equation (17). Let \( Y(S_i) \) denote the outputs associated with subsystem \( S_i \). Define

\[
[Q_{int}(s)]_{ij} = \begin{cases} Q_{ij}(s) & y_i, y_j \in Y(S_k), S_k \ \text{a subsystem in} \ \mathcal{J} \\ 0 & \text{otherwise,} \end{cases}
\]

and \( Q_{ext} \equiv \bar{Q}(s) - Q_{int}(s) \). Then the signal structure and subsystem structure are related in the following way:

\[
S \left[ \begin{array}{c|c} L & K \end{array} \right] = (I - Q_{int})^{-1} \left[ P \mid Q_{ext} \right] \tag{28}
\]

**Proof:** Examining relation (28), observe that the \( ij \)th entry of the left hand side describes the closed loop causal dependency from the \(ij\)th entry of \( [U^T \ Y^T]^T \) to \( y_i \). By closed loop, we mean that they do not describe the internal dynamics of each subsystem, e.g. the direct causal dependencies among outputs of a single subsystem. Thus, these closed loop causal dependencies are obtained by solving out the intermediate direct causal relationships, i.e. the entries in \( Q_{int} \). Notice that the right hand side of (28) also describes the closed loop map from \( [U^T \ Y^T]^T \) to \( Y \), and in particular the \( ij \)th entry of

\[
(I - Q_{int})^{-1} \left[ \bar{P} \mid Q_{ext} \right]
\]

describes the closed loop causal dependency from the \( ij \)th entry of \( [U \ Y]^T \) to \( y_i \). As a special case, notice that for SO subsystem structures, \( Q_{int} \) becomes the zero matrix and that for subsystem structures with a single subsystem, \( S \) becomes the system transfer function, \( L \) becomes the identity matrix, \( Q_{int} = \bar{Q} \), and \( Q_{ext} \) and \( K \) are both zero matrices. The primary import of this result is that a single subsystem structure can be consistent with two or more signal structures and that a single signal structure can be consistent with two or more subsystem structures. Consider the following examples:

**Example 3:** A Signal Structure consistent with two Subsystem Structures

In this example, we will show how a signal structure can be consistent with two subsystem structures. To do this we construct two different generalized state-space realizations that yield the same minimal intricacy realization but different admissible partitions (see Definition 4). The result is two different subsystem structures that are associated with the same signal structure. First, we consider the complete computational structure \( \mathcal{C}_1 \) with generalized state-space realization

\[
(A_1, B_1, C_1, D_1) = \left( \begin{array}{c|c} A_1 & \hat{A}_1 \\ \hline \hat{A}_1 & \hat{A}_1 \end{array} \right), \left( B_1 \right), \left[ C_1 \mid \bar{C}_1 \right], D_1
\]

where

\[
A_1 = \begin{bmatrix} \begin{array}{cccc} -4 & 1 & 0 & 0 \\ 1 & -7 & 0 & 0 \\ 0 & 0 & -6 & 0 \\ 1 & 2 & 0 & -10 \end{array} \end{bmatrix}, \quad \hat{A}_1 = \begin{bmatrix} \begin{array}{cccc} 0 & 0 & 2 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{array} \end{bmatrix}, \quad \bar{C}_1 = [0]_4,
\]

\[
\hat{A}_1 = \begin{bmatrix} \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array} \end{bmatrix}, \quad \bar{A}_1 = [0]_4
\]

and \( D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \). Figure 9(a) shows the computational structure \( \mathcal{C}_1 \).
resulting minimal intricacy realization for both systems is the same, implying results in two different subsystem structure representations. Nonetheless, the rearrangement by a slight rearrangement of their state-space dynamics. The rearrangement

Fig. 9. The complete computational structures of two systems that differ only vertices for the sake of visual clarity.

(a) The complete computational structure \( \mathcal{C}_1 \) with generalized state-space realization \( (A_1, B_1, C_1, D_1) \). We have omitted the self-loops on each of the \( f \) vertices for the sake of visual clarity.

\[
(A_2, B_2, C_2, D_2) = \left( \begin{bmatrix} A_2 & A_2 \\ \bar{A}_2 & \bar{A}_2 \end{bmatrix}, \begin{bmatrix} B_2 \\ \bar{B}_2 \end{bmatrix}, \begin{bmatrix} C_2 & \bar{C}_2 \end{bmatrix}, D_2 \right).
\]

The matrix entries of this generalized state-space realization are given as follows:

\[
A_2 = \begin{bmatrix}
-4 & 1 & 0 & 0 & 1 \\
1 & -7 & 0 & 0 & 3 \\
0 & 0 & -6 & 1 & 0 \\
0 & 0 & 2 & -6 & 0 \\
1 & 2 & 0 & 0 & -10
\end{bmatrix}, \quad \hat{A}_2 = \begin{bmatrix} 0 & 2 & 1 \\ 0 & 2 & 1 \\ 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \end{bmatrix}
\]

\[
\tilde{A}_2 = \tilde{A}_1, \quad \hat{A}_2 = [0]_{4\times 1},
\]

\[
B_2 = B_1 = 1_{5\times 1}, \quad \bar{B}_2 = \bar{B}_1 = 0_{4\times 1},
\]

\[
C_2 = C_1, \quad \bar{C}_2 = \bar{C}_1,
\]

and \( D_2 = D_1 \). Figure 9(b) shows the computational structure \( \mathcal{C}_2 \). The difference between these two computational structures is evident more in the subsystem structure representation of the system - note how replacing \( A_1 \) with \( A_2 \), essentially externalizes internal dynamics. The result is that \( \mathcal{C}_2 \) admits a subsystem structure \( \mathcal{S}_2 \) which divides one of the subsystems of \( \mathcal{S}_1 \) into two subsystems. This is more apparent in the LFT representations of \( \mathcal{S}_1 \) and \( \mathcal{S}_2 \); the LFT representation of \( \mathcal{S}_1 \) is given by \( \mathcal{S}(N_1, S_1) = \begin{bmatrix} 0_{4\times 1} \\ \mathcal{L}_1 \end{bmatrix} \]

\[
N_1 = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

and

\[
S_1(s) = \begin{bmatrix} S_{11}(s) & 0 & 0 \\ 0 & S_{12}(s) & 0 \\ 0 & 0 & S_{13}(s) \end{bmatrix},
\]

with \( S_{11}(s), S_{12}(s), S_{13}(s) \) given by

\[
\frac{2(s^2 + 18s + 76)}{s^3 + 21s^2 + 130s + 234} \quad \frac{s^2 + 18s + 76}{s^3 + 21s^2 + 130s + 234} \quad \frac{s^2 + 19s + 86}{s^3 + 21s^2 + 130s + 234}
\]

\[
\left( \begin{array}{ccc}
\frac{2}{s + 6} & \frac{1}{s + 6} & \frac{1}{s + 6} \\
\frac{2}{s + 6} & \frac{2}{s + 6} & \frac{1}{s + 6} \\
\frac{2}{s + 6} & \frac{2}{s + 6} & \frac{1}{s + 6}
\end{array} \right).
\]

Next consider the complete computational structure \( \mathcal{C}_2 \) with respectively.
The LFT representation of $\mathcal{F}_2(s)$ is represented as the LFT $\mathcal{F}(N_2, S_2) = \{N_2, S_2\}$ where

$$N_2 = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

and $S_2(s), S_{21}(s), S_{22}(s)$ respectively given as

$$S_2(s) = \begin{bmatrix} S_{21}(s) & 0 \\ 0 & S_{22}(s) \end{bmatrix},$$

$$S_{21}(s) = \begin{bmatrix} \frac{2(s^2+18s+76)}{s^3+21s^2+130s+234} & \frac{s^2+18s+76}{2s^3+15s+52} \\ \frac{s^2+18s+76}{s^3+21s^2+130s+234} & \frac{s^2+19s+86}{s^3+21s^2+130s+234} \end{bmatrix},$$

$$S_{22}(s) = \begin{bmatrix} \frac{2s^3+12s^2+34}{s^3+12s^2+34} & \frac{s^3+18s+76}{2s^3+15s+52} \\ \frac{s^3+12s^2+34}{s^3+12s^2+34} & \frac{s^3+21s^2+130s+234}{s^3+21s^2+130s+234} \end{bmatrix}.$$

However, if we consider the minimal intrinsic realizations of $\mathcal{G}_1, \mathcal{G}_2$ we get the same state-space realization $(A_o, B_o, C, D_o)$ with

$$A_o = \begin{bmatrix} -4 & 1 & 2 & 1 & 1 \\ 1 & -7 & 2 & 1 & 3 \\ 2 & 1 & -6 & 1 & 0 \\ 1 & 2 & 2 & -6 & 0 \\ 1 & 2 & 0 & 0 & -10 \end{bmatrix}, \quad B_o = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} I_4 & 0_{4 \times 1} \end{bmatrix}.$$

The signal structure of the system is thus specified by the dynamical structure function $(Q, P)(s)$, with

$$Q(s) = \begin{bmatrix} 0 & \frac{12+x}{s^3+12x+39} & \frac{2(x+10)}{s^3+14x+39} & \frac{x+10}{s^3+x+39} \\ \frac{13+x}{s^3+17x+64} & 1 & \frac{2(s+10)}{s^3+17x+64} & \frac{x+10}{s^3+17x+64} \\ \frac{1}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \end{bmatrix}$$

and

$$P(s) = \begin{bmatrix} \frac{11+x}{s^3+x+39} & \frac{13+x}{s^3+17x+64} \\ \frac{1}{s+6} & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{1}{s+6} \end{bmatrix}.$$

*Example 4: A Subsystem Structure consistent with two Signal Structures*

Now we consider a subsystem structure with multiple signal structures. Recalling the discussion above, subsystem structure describes the *closed loop* causal dependencies between manifest interconnection variables and signal structure specifies direct causal dependencies between manifest variables. When it is impossible to determine these direct causal dependencies by inspection from the closed loop causal dependencies in a subsystem structure representation, then there can be multiple signal structures that are consistent with the subsystem structure.

Reconsider $\mathcal{F}_2$. The LFT is given by $\mathcal{F}(N_2, S_2)$ where

$$N_2 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix},$$

and

$$S_2 = \begin{bmatrix} S_{21} & 0 \\ 0 & S_{22} \end{bmatrix},$$

with $S_{21}$ and $S_{22}$ given by

$$S_{21}(s) = \begin{bmatrix} \frac{2(s^2+18s+76)}{s^3+21s^2+130s+234} & \frac{s^2+18s+76}{2s^3+15s+52} \\ \frac{s^2+18s+76}{s^3+21s^2+130s+234} & \frac{s^2+19s+86}{s^3+21s^2+130s+234} \end{bmatrix},$$

$$S_{22}(s) = \begin{bmatrix} \frac{2s^3+12s^2+34}{s^3+12s^2+34} & \frac{s^3+18s+76}{2s^3+15s+52} \\ \frac{s^3+12s^2+34}{s^3+12s^2+34} & \frac{s^3+21s^2+130s+234}{s^3+21s^2+130s+234} \end{bmatrix}.$$

respectively.

If we consider the relation

$$S_2[K \quad L] = (I - Q_{net})^{-1} \begin{bmatrix} Q_{ext} & P \end{bmatrix},$$

we can take $(Q, P)(s)$ to equal

$$Q_1(s) = \begin{bmatrix} 0 & \frac{12+x}{s^3+12x+39} & \frac{2(s+10)}{s^3+14x+39} & \frac{x+10}{s^3+x+39} \\ \frac{13+x}{s^3+17x+64} & 1 & \frac{2(s+10)}{s^3+17x+64} & \frac{x+10}{s^3+17x+64} \\ \frac{1}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \end{bmatrix},$$

and

$$P_1(s) = \begin{bmatrix} \frac{11+x}{s^3+x+39} & \frac{13+x}{s^3+17x+64} \\ \frac{1}{s+6} & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{1}{s+6} \end{bmatrix}.$$
and \( Q_{ext} \equiv Q_1 - Q_{intr} \), or \( Q_2(s) = \)
\[
\begin{bmatrix}
0 & 0 & \frac{2(s^2 + 18s + 76)}{s^2 + 21s^2 + 130s + 234} & \frac{s^2 + 18s + 76}{s^2 + 21s^2 + 130s + 234} \\
0 & 0 & \frac{2(52s + 15s + s^2)}{s^2 + 21s^2 + 130s + 234} & \frac{52s + 15s + s^2}{s^2 + 21s^2 + 130s + 234} \\
\frac{2s + 13}{s^2 + 12s + 34} & \frac{s^2 + 8}{s^2 + 12s + 34} & 0 & 0 \\
\frac{s + 10}{s^2 + 12s + 34} & \frac{2(7s + 1)}{s^2 + 12s + 34} & 0 & 0
\end{bmatrix}
\]

and taking \( Q_{intr} \equiv [0] \) and \( Q_{ext} \equiv Q_2 \). It is routine to show that with these definitions, both \((Q_1, P_1)(s)\) and \((Q_2, P_2)(s)\) are consistent with \( S_2 \). Thus a single subsystem structure can be consistent with two signal structures. It is important to note that one of our signal structures exhibited direct causal dependencies corresponding exactly with the closed loop dependencies described by subsystem structure. We note that this correspondence sometimes occurs (as in the case of Theorem 3) but generally does not hold.

V. IMPACT ON SYSTEMS THEORY

The introduction of partial structure representations suggests new problems in systems theory. These problems explore the relationships between different representations of the same system, thereby characterizing properties of the different representations. For example, classical realization theory considers the situation where a system is specified by a given transfer function, and it explores how to construct a consistent state space description. Many important ideas emerge from the analysis:

1) State realizations are generally more informative than a transfer function representation of a system, as there are typically many state realizations consistent with the same transfer function.

2) Order of the state realization is a sensible measure of complexity of the state representation, and there is a well-defined minimal order of any realization consistent with a given transfer function; this minimal order is equal to the Smith-McMillian degree of the transfer function.

3) Ideas of controllability and observability of a state realization characterize important properties of the realization, and any minimal realization is both controllable and observable.

In a similar way, introducing partial structure representations impacts a variety of concepts in systems theory, including realization, minimality, and model reduction.

A. Realization

The definition of partial structure representations enrich the kinds of realization questions one may consider. In the previous section, we demonstrated that partial structure representations of a system are generally more informative than its transfer function but less informative than its state realization. Thus, two classes of representation questions emerge: reconstruction problems and structure realization problems (Figure 10).

Reconstruction problems consider the construction of a partial structure representation of a system given its transfer function. Because partial structure representations are generally more informative than a transfer function, these problems are ill-posed-like the classical realization problem. Nevertheless, one may consider algorithms for generating canonical representations, or one may characterize the additional information about a system—beyond knowledge of its transfer function—that one would require in order to specify one of its partial structure representations. In particular, we may consider two types of reconstruction problems:

1) **Signal Structure Reconstruction**: Given a transfer function \( G(s) \) with its associated sparsity structure \( \mathcal{Z} \), find a signal structure, \( \mathcal{W} \), with dynamical structure function \((Q, P)\) as in Equation (17) such that \( G = (I - Q)^{-1}P \).

2) **Subsystem Structure Reconstruction**: Given a transfer function \( G(s) \) with its associated sparsity structure \( \mathcal{Z} \), find a subsystem structure, \( \mathcal{J} \), with structured LFT \((N, S)\) as in Equation (11) such that \( G = (I - SK)^{-1} SL \).

Signal structure reconstruction is also called network reconstruction, particularly in systems biology where it plays a central role. There, the objective is to measure fluctuations of various proteins, or other chemical species, in response to particular perturbations of a biochemical system, and then infer causal dependencies among these species [31], [32], [29], [24], [33], [34].

Structure realization problems then consider the construction of a state space model, possibly generalized to include auxiliary variables as necessary, consistent with a given partial structure representation of a system. Like the classical realization problem or reconstruction problems, these problems are also ill-posed since there are typically many state realizations of a given partial structure representation of a system. Also, like reconstruction problems, these realization problems can be categorized in two distinct classes, depending on the type
of partial structure representation that is given:

3) **Signal Structure Realization:** Given a system \( G \) with signal structure \( \mathcal{W} \) and associated dynamical structure function \((Q, P)\), find a state space model \((A, B, C, D)\) consistent with \((Q, P)\), i.e. such that Equations (14-17) hold.

4) **Subsystem Structure Realization:** Given a system \( G \) with subsystem structure \( \mathcal{S} \) and associated structured \((N, S)\) (recorded in the LFT representation \( \mathcal{F}(N, S) \)), find a generalized state space model of the form of Equation (5) consistent with \((N, S)\).

Signal structure realization may sometimes be called *network realization*, consistent with the nomenclature for signal structure reconstruction.

Note that all the reconstruction and structure realization problems here are distinct from identification problems, just as classical realization is different from identification. For the systems considered here, identification refers to the use of input-output data (and no other information about a system) to choose a representation that best describes the data in some appropriate sense. Because input-output data only characterizes the input-output map of a system, identification can at best characterize the system’s transfer function; no information about structure, beyond the sparsity structure, is available in such data. In spite of this distinction, however, it is not uncommon for reconstruction problems to be called *structure identification* problems. Nevertheless, one may expose such problems as the concatenation of an identification and a reconstruction problem and precisely characterize the extra information needed to identify such structure by carefully analyzing the relevant reconstruction problem, independent of the particular identification technique [24], [31].

### B. Minimality

Just as partial structure representations enrich the classical realization problem, they also impact the way we think about minimality. Certainly the idea of a minimal complexity representation is relevant for each of the four problems listed above, but clearly the relevant notion of complexity may be different depending on the representation. We consider each situation as follows:

1) **Minimal Signal Structure Reconstruction:** In this situation one needs to consider how to measure the complexity of a system’s signal structure, \( \mathcal{W} \), or its associated dynamical structure function, \((Q, P)\). Some choices one may consider could be the number of edges in \( \mathcal{W} \), the maximal order of any element in \((Q, P)\), or the maximal order of any path from any input \( u_i \) to any output \( y_j \). The problem would then be to find a minimal complexity signal structure consistent with a given transfer function.

2) **Minimal Subsystem Structure Reconstruction:** In this situation one needs to consider how to measure the complexity of a system’s subsystem structure, \( \mathcal{S} \), or its associated structured LFT, \((N, S)\). One notion could be to measure complexity by the number of distinct subsystems; the problem would then be to find the minimal complexity subsystem structure representation consistent with a given transfer function. Another notion could be the number of non-zero entries in the \( S \) matrix, where \((N, S)\) denote the the LFT associated with the subsystem structure. Using this measure, a single subsystem block with no zero entries would be considered a more complex representation than a subsystem structure with a large number of distinct, albeit interconnected, subsystems.

3) **Minimal Signal Structure Realization:** In this situation one needs to consider how to measure the complexity of a system’s zero-intricacy state realization, from which signal structure is derived. The obvious choice would be to use the order of the realization as a natural measure of complexity, and the problem would then be to find the minimal order state realization [29], [35] consistent with a given signal structure, \( \mathcal{W} \), or, equivalently, with its associated dynamical structure function, \((Q, P)\). Note that this minimal order is guaranteed to be finite (for the systems considered here) and can easily be shown to be greater or equal to the Smith-McMillian degree of the transfer function specified by the signal structure; we call this number the *structural degree* of the signal structure [25].

4) **Minimal Subsystem Structure Realization:** In this situation one needs to consider how to measure the complexity of a generalized state realization in the presence of auxiliary variables. Both the order and the intricacy of the realization offer some perspective of its complexity, but one needs to consider how they each contribute to the overall complexity of the realization. The problem would then be to find a minimal complexity generalized state realization consistent with a given subsystem structure.

These various problems demand new ideas for thinking about the complexity of a system’s representation, especially that of a partial structure representation. These new ideas about complexity, in turn, introduce opportunities for characterizing minimality of a representation in terms that add insight to our understanding of the relationship between a system’s behavior and its structure, much as controllability and observability characterize classical notions of minimality in a system’s state realization. Besides suggesting the need for a characterization of minimality, however, these ideas also impact notions of approximation and how we think about model reduction.

### C. Model Reduction

Each of the reconstruction and structure realization problems described above have associated with them not only a minimal-representation version of the problem, but also an approximate-representation version of the problem. The minimal-representation versions of these problems, as described above, seek to construct a representation of minimal complexity in the targeted class that is nevertheless consistent.
with the system description provided. Similarly, approximate-representation versions of these problems seek to construct a representation in the targeted class that has a lower complexity than the minimal complexity necessary to deliver consistency with the system description provided. As a result, consistency with the given system description can not be achieved, so measures of approximation become necessary to sensibly discuss a “best” representation of the specified complexity.

For example, associated with the classical realization problem is the standard model reduction problem. In this situation, a transfer function is specified, and one would like to construct a state realization with a complexity that is lower than that which is necessary (for such a realization to be consistent with the given transfer function) that nevertheless “best” approximates it. Note that the appropriate notion of complexity depends on the target representation class; here, the target representation class is the set of state realizations, so the appropriate notion of complexity is model order. Likewise, note that the appropriate notion of approximation depends on the type of system representation that is initially provided; here, a transfer function is provided, so an appropriate measure of approximation could be an induced norm, such as $H_\infty$. Thus, one could measure the quality of an approximation by measuring the induced norm of the error between the given transfer function and that specified by the approximate state realization. Note that since the $H_\infty$ model reduction problem remains open, many alternative measures and approaches have been suggested [36], [37], [23]. In any event, because the specified system description is a transfer function, the resulting measure of approximation is typically one that either directly or indirectly measures the difference in input-output dynamic behavior between the approximate model and the given system description; the focus is on dynamics, not system structure, when considering notions of approximation in the standard model reduction problem.

Similarly, there is an appropriate reduction problem associated with each of the minimal reconstruction and minimal structure realization problems described above. Like standard model reduction, the appropriate notion of complexity is characterized by the class of target representations, while the appropriate measure of approximation depends on the system representation that is initially provided. To make these ideas more concrete, we discuss each of the problems individually:

1) **Approximate Signal Structure Reconstruction**: In this situation one would like to find a signal structure representation with lower complexity e.g. fewer number of edges, etc. than the minimal level of complexity necessary to specify a given transfer function. When using the number of edges as a complexity measure, this problem may be interpreted as trying to restrict the number of communication links between individual computation nodes in a distributed system while approximating some desired global dynamic. The appropriate measure of approximation, then, is any measure that sensibly compares the desired transfer function from that specified by the reduced signal structure, such as $H_\infty$.

2) **Approximate Subsystem Structure Reconstruction**: In this situation one would like to find a subsystem structure representation with lower complexity than the minimal level of complexity necessary to specify a given transfer function. If one considers the number of subsystems as the measure of complexity, then this problem is trivial since any transfer function is consistent with a subsystem structure with a single subsystem. If one measures complexity by the number of non-zero entries in the $S$ matrix, where $(N,S)$ denote the the LFT associated with the subsystem structure, then it appears likely that one could often formulate a meaningful approximation problem.

Unlike these approximate reconstruction problems, approximate realization problems may have dual relevant measures of approximation, since having an initial partial structure representation of a system also always specifies its transfer function. Measuring the similarity between transfer functions determines the degree to which a lower order system approximates the dynamics of a given system, while measuring the similarity between partial structure representations determines the degree to which a lower order system approximates the structure of a given system.

3) **Approximate Signal Structure Realization**: In this situation one would like to find a state realization with a model complexity that is lower than the minimal complexity necessary to specify a given signal structure. Typically the complexity of a generalized state realization would be measured by both the order and intricacy of the realization. Since signal structure only depends on the zero intricacy realization of a system, however, a minimal complexity realization will always have zero intricacy; thus model order is the only relevant notion of complexity. Moreover, since the structural degree of a particular signal structure may be strictly greater than the Smith-McMillian degree of the transfer function it specifies, a few distinct kinds of approximation problems emerge: structural approximation and dual approximation (Figure 11).

a) **Structural Approximation**: When the order of the approximation is less than the structural degree of the given signal structure but greater than the Smith-McMillian degree of the associated transfer function, the resulting approximation can specify the given transfer function exactly, even though its signal structure only approximates that of the given system. Note that the sense in which similarity in signal structure should be measured is an area of on-going research.

b) **Dual Approximation**: When the order of the approximation is less than the Smith-McMillian degree of the transfer function specified by the given signal structure, then it will represent neither the structure nor the dynamics of the given system exactly.
4) **Approximate Subsystem Structure Realization**: In this situation one would like to find a generalized state realization with a model complexity that is lower than the minimal complexity necessary to specify a given subsystem structure. Here, complexity of a generalized state realization is measured both in terms of intricacy and order since intricacy of a realization directly impacts the number of admissible subsystem blocks (see Figure 3) while order impacts the ability of the realization to approximate the transfer function specified by the given subsystem structure. As a result, three distinct approximation problems emerge to complement subsystem structure realization: Structure-Preserving Model Reduction, Subsystem Structure Approximation, Subsystem Dual Approximation (Figure 12).

   a) **Structure-Preserving Model Reduction**: When the intricacy of an approximation is high enough, the subsystem structure of a system can be preserved while its dynamic behavior is approximated by lower-order systems. A naive approach to such reduction would be to reduce the order of various subsystems. However, even when each subsystem is well approximated, the dynamic behavior of the overall interconnection may be very different from the original system. As a result, methods have been developed in this area for approximating the dynamics of the closed-loop, interconnected system while preserving its subsystem structure [38], [39], [40].

   b) **Subsystem Structure Approximation**: When the order of an approximation is high enough, the dynamic behavior of a system can be preserved while its subsystem structure is approximated by lower-intricacy realizations. The sense in which similarity of subsystem structure should be measured remains an open topic for research.

   c) **Subsystem Dual Approximation**: When both the order and the intricacy of an approximation are lower than the minimal values necessary to realize a given subsystem structure and the transfer function it specifies, then the objective of the reduction problem is to find the realization of the specified complexity that best approximates the structure and dynamics of the given system.

The introduction of partial structure representations suggests a number of new problems in systems theory. These problems include new classes of realization problems, called reconstruction and structural realization problems, as well as a number of new reduction problems. Each of these problems differ depending on the partial structure representation one considers, and a number of research issues remain to properly formulate most of them. The overview offered here is merely meant to give a perspective of the landscape of problems that emerges with the introduction of partial structure representations.

**VI. Conclusion**

This paper introduced the idea of a system’s complete computational structure as a baseline for comparing simplified representations. Although closely aligned with a system’s state
space realization, we demonstrated the need for auxiliary variables to encode information about a system’s admissible subsystem structure. This results in a “generalized” state description that is a differential algebraic system with differentiation index zero, and the number of auxiliary variables becomes an additional measure of complexity (besides model order) that we call intricacy. This generalized state representation, and its associated graphical representation, is the system’s complete computational structure.

Partial structure representations were then introduced as a means for simplifying a system’s structural description while retaining a complete representation of its input-output dynamic behavior. These included subsystem structure, signal structure, and sparsity structure.

Subsystem structure was first introduced as the most refined view of the interconnection of a system’s legitimate subsystems. This description is represented as the lower linear fractional transformation of a static interconnection matrix $N$ with a block diagonal operator $S$. Its graphical representation is a block diagram, like the system’s complete computational structure, that is a condensation graph with respect to a meaningful partition of the system’s states.

Signal structure, on the other hand, is a signal flow diagram of the causal dependencies among manifest variables given by the dynamical structure function of the system. We demonstrated that systems may exhibit extremely structured behavior in the signal structure sense while having no apparent structure in the subsystem or computational structure sense. Moreover, the transformations between manifest variables, represented as edges in the signal structure graph, do not necessarily partition the system states, as do the nodes of the subsystem structure. This fact implies that the minimal order to realize a particular signal structure may be, in fact, higher than the minimal order necessary to realize the transfer function specified by the given signal structure.

Finally, the weakest notion of structure is sparsity structure, or the pattern of zero entries in the system’s transfer function matrix, which graphically also becomes a signal flow graph like signal structure. We demonstrated that this representation is very weak, reminding readers that a diagonal transfer function matrix, for example, does not imply that even a minimal realization of the system is necessarily decoupled. Thus, this notion of structure really is a statement about the closed-loop dependencies of inputs on outputs of the system and much less of a statement about how those closed-loop dependencies came to be.

These representations were then shown to contain differing levels of structural information about a system. In particular, it was shown that the complete computational structure uniquely specifies both the subsystem and signal structure of a system, and that either of these partial structure representations uniquely specify the transfer function (and thus its associated sparsity structure) of the system. Nevertheless, the relationship between subsystem and signal structure is less definitive, as we demonstrated that two realizations of the same system may share subsystem structure but have different signal structures, or, conversely, two realization of the same system may share signal structure but have different subsystem structures. These different representations simply appear to offer different perspectives of the system’s structural properties.

We then surveyed the landscape of new problems in systems theory that arise when considering these partial structure representations. In particular, we showed that the standard realization problem now becomes two types of problems, a reconstruction problem, where a transfer function is given and one would like to determine a partial structure representation that is consistent with it, and a structure realization problem, where a partial structure representation is given and one would like to find a generalized state realizations that is consistent with it. Minimal versions of these problems are obtained if one can define a sensible notion of complexity of each kind of representation, and various suggestions were offered. Associated approximation, or model reduction problems were then characterized, where the target representation is simpler than the minimal complexity necessary to yield a representation that is consistent with the given model or description of the system. Here we note that the model reduction problems begin to consider structure approximation as well as approximation of the system’s dynamic behavior, leading to a variety of new problems one may consider. It is our hope that many of these problems will be addressed in the coming years, leading to a more thorough and complete understanding of the meaning of structure and its relationship to the behavior of interconnected dynamic systems.

VII. ACKNOWLEDGMENTS

This work was supported in part by a Charles Lee Powell Foundation Fellowship, the Air Force Research Laboratory Grant FA 8750-09-2-0219 and by the Engineering and Physical Sciences Research Council Grant EP/G066477/1.

REFERENCES

[1] H. Nyquist, “Regeneration theory”, Bell System Technical Journal, vol. 11, pp. 126–147, September 1932.
[2] H. S. Black, “Stabilized feedback amplifiers”, Bell System Technical Journal, vol. 13, pp. 1–18, 1934.
[3] H. W. Bode, “Relations between attenuation and phase in feedback amplifier design”, Bell System Technical Journal, vol. 19, pp. 421–454, 1940.
[4] J. C. Maxwell, “On governors”, Proceedings of the Royal Society, vol. 16, no. 100, pp. 270–283, 1868.
[5] H. Raza and P. Ioannou, “Vehicle following control design for automated highway systems”, Control Systems Magazine, vol. 16, no. 6, December 1996.
[6] J. Fowler and R. D’Andrea, “A formation flight experiment”, Control Systems Magazine, October 2003.
[7] R. D’Andrea and G. Dullerud, “Distributed control for spatially interconnected systems”, IEEE Transactions on Automatic Control, vol. 48, no. 9, pp. 1478–1496, September 2003.
[8] D. Del Vecchio, A. J. Ninfa, and E. D. Sontag, “Modular cell biology: retroactivity and insulation”, Molecular Systems Biology, vol. 4, no. 161, 2008.
[9] S. Jayanthi and D. Del Vecchio, “On the compromise between retroactivity attenuation and noise amplification in gene regulatory networks”, Proceedings of the 2009 IEEE Conference on Decision and Control, December 2009.
[10] D. Luenberger, “Time-invariant descriptor systems”, Automatica, vol. 14, pp. 473–480, 1978.
