Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 12015–12020

Distributed/Hierarchical Control Architecture Design ⋆

Wentao Tang, Prodromos Daoutidis *

* *Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, Minnesota 55455, USA (e-mail: tangx647@umn.edu, daout001@umn.edu).*

Abstract: Distributed and/or hierarchical control strategies occupy a central role in controlling complex networks effectively. A systematic decomposition method for designing the corresponding architectures is currently lacking. In this paper, we propose a network theoretic method to design distributed/hierarchical control architectures. The system digraph is first simplified through unimportant edge removal and node agglomeration. The connected components of the simplified network form a subsystem in a distributed control architecture. For each subsystem, hierarchical input-output subnetworks are detected using the Pichai-Sezer-Šiljak algorithm, thus generating a hierarchical architecture. The proposed method is illustrated through a reaction-separation process.

Keywords: Control architecture design, hierarhical control, distributed control.

1. INTRODUCTION

Large-scale systems with complex interconnections, including biomolecular reactions (Strogatz, 2001), electric grids (Farhangi, 2010), and modern chemical and energy plants (Baldea and Daoutidis, 2012), typically exhibit a networked structure. Hierarchical and distributed control strategies, such as hierarchical and distributed model predictive control (see Scattolini (2009); Christofides et al. (2013) for excellent reviews and reference lists), play a central role in the control of such networked systems. In distributed control, through information sharing between subsystems, better performance can in principle be achieved compared to decentralized controllers. On the other hand, hierarchical control architectures organize the controllers in a multilayer pattern, with transfer of information across different layers of the hierarchy; such architectures have been applied to systems with different time scales, e.g. networks with large recovery and recycle of material or energy, where network-level dynamics are slow and unit-process dynamics are fast (Baldea and Daoutidis, 2014; Jogwar et al., 2015).

A major open problem in this area is the systematic decomposition of a networked system into the distributed or hierarchical architecture. Early large-scale system decomposition studies include strongly connected states agglomeration (Callier et al., 1976; Vidyasagar, 1980), hierarchical input-output reachable subsystem decomposition, (Pichai et al., 1983), lower block triangular rearrangement (Šiljak, 1991) and nested ε-decomposition (Sezer and Šiljak, 1986). However, the special network structure or matrix coefficient conditions imposed in these works are usually restrictive. More recent approaches use optimization formulations (Groß and Stursberg, 2011; Jilg and Stursberg, 2013). These methods, however, are largely limited to proportional feedback control of linear systems, and the resulting mixed integer semi-definite optimization problems are computationally demanding.

Developments in network theory offer the perspective to consider the network decomposition as a *community detection* problem, which aims to categorize the nodes of a network into subsets (communities), such that the interconnections inside communities are significantly stronger than the interconnections between communities (Girvan and Newman, 2002; Fortunato, 2010). An abundance of computationally efficient community detection algorithms have been developed and made available (Fortunato and Hric, 2016). The use of community detection in the context of control-relevant decomposition has only recently been pursued. Specifically, structural connectivity was used as an interaction measure between inputs and outputs, and hierarchical agglomerative and divisive clustering approaches were adopted (Heo et al., 2015; Heo and Daoutidis, 2016; Kang et al., 2016) to generate weakly interacting input-output subsystems with strongly interacting variables inside, thus providing control architectures suitable for distributed control.

In this paper, we propose a method of designing hybrid distributed/hierarchical control architectures. The method involves the identification of weakly interacting subnetworks through a simplification of the network by unimportant edge removal, followed by a decomposition of each subnetwork into hierarchical structures using acyclic input-output reachable partition with the Pichai-Sezer-Šiljak algorithm (Pichai et al., 1983). The method generates structurally controllable and observable subsystems organized in hierarchical structures within a distributed structure. The proposed method is well-suited for many real-world networks with an underlying modular structure where the corresponding subnetworks are complex them-

⋆ Financial support from NSF-CBET is gratefully acknowledged.

10.1016/j.ifacol.2017.08.2128

selves. A reactor-separator process is used as an illustrating example.

2. NETWORK DECOMPOSITION METHOD

In the following, we consider a nonlinear system:

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned} \tag{1}$$

in which $x \in \mathbb{R}^n, u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$ are state variables, inputs and outputs, respectively. The equations are translated so that the origin is the operation point (an equilibrium point under zero input), i.e. $f(0,0) = 0$.

We refer as the *system digraph* the network in which the nodes represent process variables (inputs u_i, states x_k, and outputs y_j), and there is an edge from u_i to x_k if $\partial f_k/\partial u_i \not\equiv 0$, from x_k to x_l if $\partial f_l/\partial x_k \not\equiv 0$, from x_k to y_j if $\partial h_j/\partial x_k \not\equiv 0$, and from u_i to y_j if $\partial h_j/\partial u_i \not\equiv 0$. We assume that for any edge in the system digraph, the corresponding function ($\partial f_k/\partial u_i, \partial f_l/\partial x_k$, $\partial h_j/\partial x_k$ and $\partial h_j/\partial u_i$) is nonzero at the origin.

In order to design the control architecture, we first consider reducing the complexity of the network by removing the edges which are not important in a certain sense. The simplification we seek is two-fold. First, if the network possesses a modular structure with weakly interacting communities, we seek to identify these communities and decompose the network into disconnected components by ignoring these weak intra-community interactions. This naturally suggests a distributed control architecture wherein these communities are the different controller subsystems. Second, we seek to eliminate cycles in the network, caused by recycle structures, as this may help reveal possible hierarchical structures. The detection of such hierarchical structures follows the simplification.

2.1 Distributed control architecture design

We begin by assigning a weight (length) to each edge on the system digraph as follows (Tang and Daoutidis, 2016):

$$\begin{aligned} w(u_i,x_k) &= 1 - \log_{10}|\partial f_k(0,0)/\partial u_i|, \\ w(x_k,x_l) &= 1 - \log_{10}|\partial f_l(0,0)/\partial x_k|, \\ w(x_k,y_j) &= -\log_{10}|\partial h_j(0,0)/\partial x_k|, \\ w(u_i,y_j) &= -\log_{10}|\partial h_j(0,0)/\partial u_i|. \end{aligned} \tag{2}$$

The edge weights comprise of two terms. The first term accounts for the dynamic effect, assigned to be 1 in the presence of a differential equation (for input-state and state-state edges) and 0 otherwise (for state-output and input-output edges). The second term accounts for the response sensitivity, according to the corresponding coefficient of the linearized system. The use of $-\log_{10}$ negatively correlates the weights to the sensitivities and allows expressing the composite effect of an input on an output through the lengths of input-output paths, by relating the multiplication of sensitivities to the summation of weights. The input-output path lengths obtained thus combine the response directness and sensitivity, and capture the short-time input-output response intensity (Tang and Daoutidis, 2016). On the weighted system digraph, one can apply shortest path search to obtain three types of input-output path lengths:

- the shortest path length between an u_i and an output y_j, L_{ij};
- the shortest path from u_i to y_j passing through a node v, $L_{ij}(v)$;
- the shortest path from u_i to y_j containing an edge e, $L_{ij}(e)$.

We now define *betweenness* measures based on the different shortest input-output paths to characterize the *un*importance of each edge. This idea and the term betweenness have their roots in the classical Girvan-Newman algorithm for community detection (Girvan and Newman, 2002), wherein the recursive removal of likely inter-community edges is applied to generate communities. Specifically, for each edge we want to characterize whether it is important for the input-output interactions or for the head and tail nodes incident to this edge. We thus define an *input/output betweenness* β^{io} for an edge e as

$$\beta^{\mathrm{io}}(e) := \min_{i,j}[L_{ij}(e) - L_{i\cdot} - L_{\cdot j} + L_{ij}], \tag{3}$$

in which $L_{i\cdot} = \min_j L_{ij}$ is the length of the shortest input-output path starting at u_i, and $L_{\cdot j} = \min_i L_{ij}$ is the length of the shortest input-output path ending at y_j. Note that β^{io} results from the differences (i) between $L_{ij}(e)$ and L_{ij}, (ii) between L_{ij} and $L_{i\cdot}$, and (iii) between L_{ij} and $L_{\cdot j}$:

$$\beta^{\mathrm{io}}(e) = \min_{i,j}[(L_{ij}(e) - L_{ij}) + (L_{ij} - L_{i\cdot}) + (L_{ij} - L_{\cdot j})]. \tag{4}$$

Hence a small β^{io}, or a high importance of e, requires that e is contained in a short path between an input u_i and an output y_j, such that y_j is close to u_i and u_i is close to y_j. The *head/tail betweenness* β^{ht} for an edge $e = (v,w)$ is defined as

$$\beta^{\mathrm{ht}}(e) := \min\{L_{\cdot\cdot}(e) - L_{\cdot\cdot}(v), L_{\cdot\cdot}(e) - L_{\cdot\cdot}(w)\}, \tag{5}$$

in which $L_{\cdot\cdot}(v) = \min_{i,j} L_{ij}(v)$ is the length of the short input-output path passing through node v (the same for w), and $L_{\cdot\cdot}(e) = \min_{i,j} L_{ij}(e)$ is the length of the shortest input-output path containing the edge e. A small β^{ht} indicates that the e is contained in a short input-output path passing through v (or w), and hence is important for its head and tail nodes. An illustration of betweenness is given in Figure 1.

Finally, we define the *betweenness* of an edge e as

$$\beta(e) = \min\{\beta^{\mathrm{io}}(e), \beta^{\mathrm{ht}}(e)\}. \tag{6}$$

If an edge e has both large β^{io} and large β^{ht}, i.e., is unimportant both with regard to the inputs and the outputs, and with regard to its head and tail nodes, then the edge can be removed. By setting different thresholds of betweenness for edge removal, networks with different extents of simplification can be obtained. The maximum simplification is obtained when all edges with $\beta(e) > 0$ are removed. In this case, an edge remains in the network after simplification only if the edge is either on the shortest input-output path between an input u_i and an output y_j such that u_i is the closest input to y_j and y_j is the closest output to u_i, or on the shortest input-output path passing through its head or its tail node.

Through this simplification procedure, the network, if possessing a modular structure, will become disconnected due to the removal of the edges between weakly interacting communities. This suggests a distributed control architecture, wherein each connected component of the simplified