Computers and Chemical Engineering 178 (2023) 108382

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/cace

Automatic decomposition of large-scale industrial processes for distributed MPC on the Shell–Yokogawa Platform for Advanced Control and Estimation (PACE)

Wentao Tang [a,*], Pierre Carrette [b], Yongsong Cai [b], John M. Williamson [b], Prodromos Daoutidis [c]

[a] Department of Chemical and Biomolecular Engineering, North Carolina State University, Raleigh, NC 27695, USA
[b] Surface Operations, Projects and Technology, Shell Global Solutions (U.S.) Inc., Houston, TX 77082, USA
[c] Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA

ARTICLE INFO

Keywords:
Network decomposition
Community detection
Model predictive control
Plantwide control

ABSTRACT

The kernel of industrial advanced process control (APC) lies in the formulation and solution of model predictive control (MPC) problems, which specify the controller moves according to the solution of an optimal control problem at each sampling time. A significant challenge is the online computation for large-scale industrial systems. As the state-of-the-art APC technology, the Shell–Yokogawa Platform for Advanced Control and Estimation (PACE) has adopted a systematic framework of handling dynamic optimization of large-scale systems, where an automatic decomposition procedure generates subsystems for distributed MPC. The decomposition is implemented on network representations of the MPC models that capture interactions among process variables, with community detection used to maximize the statistical significance of the subnetworks with preferred internal interconnections. This paper introduces the fundamentals of such a decomposition approach and this functionality in PACE, followed by a case study on a crude distillation process to showcase its industrial application.

1. Introduction

The deployment of advanced process control (APC) in the chemical process industry since the 1970s has brought significant benefits (Qin and Badgwell, 2003). An APC platform adopts a model predictive control (MPC) formulation to handle multi-input–multi-output constrained systems. That is, the control decisions are made according to the solution of an optimal control problem, where a cost function is minimized subject to model dynamics and constraints (Rawlings et al., 2017). As such, APC allows stable process operations, increases economic profits, reduces carbon emissions, and decreases the frequency of alarms and operator intervention. Shell's heralding APC technology, from Dynamic Matrix Control (DMC) (Cutler and Ramaker, 1979), Quadratic Dynamic Matrix Control (QDMC) (Garcia and Morshedi, 1986), Shell Multivariable Optimizing Control (SMOC) (Marquis and Broustail, 1988), SMOCPro (Cott, 2007), to the Platform for Advanced Control and Estimation (PACE) developed in alliance with Yokogawa (Amrit et al., 2015), has been continually evolving with advances in computing and software capabilities.

The integration of mass and energy in process designs results in the common existence of large-scale chemical plants comprising of interconnected units, also referred to as process networks (Baldea and Daoutidis, 2012). A key challenge to the control of such large-scale processes is that a centralized paradigm of optimizing over the monolithic system is undesirable, either due to the computational time needed by the optimization solver or due to its inflexibility. On the other hand, simply partitioning the system into multiple parts and controlling each of them as if they do not interact with each other (decentralized control) usually will not retain the benefits of a system-wide MPC. Therefore, *decomposition and coordination* are needed to achieve large-scale optimal control decisions on the basis of *interacting subsystems*. This idea can be traced back to Morari et al. (1980); in recent research, this frequently goes under the name of "*distributed MPC*" (Scattolini, 2009; Christofides et al., 2013).

We also remark that recently, pre-training neural networks as an explicit MPC solver (Katz et al., 2020; Chen et al., 2022) has been considered as a useful approach to ease the online dynamic optimization. The work of Kumar et al. (2021) applied neural-explicit MPC to a process at realistic industrial scale with orders-of-magnitude computational acceleration. It will be of interest to investigate the synergy between distributed control (which remains model-based, requires little tuning, and keeps the deployment workflow) and data-driven approaches

* Corresponding author.
E-mail addresses: wentao_tang@ncsu.edu (W. Tang), daout001@umn.edu (P. Daoutidis).

https://doi.org/10.1016/j.compchemeng.2023.108382
Received 25 May 2023; Received in revised form 2 August 2023; Accepted 3 August 2023
Available online 7 August 2023

(which is semi-model-based or model-free, depends strongly on hyperparameters, and tends to restructure the control technology) (Tang and Daoutidis, 2022a).

Two key questions to the implementation of distributed MPC are *how to decompose the process model* and *how to coordinate the subsystem controllers*. These two aspects are tightly interconnected and have a complex effect on the computational and control performance of the system, which are overall open questions for future investigations. Ideally, the coordination scheme should be designed in such a way that guarantees the convergence to the monolithic optimum regardless of the decomposition configuration, provided sufficiently many iterations among the distributed controllers. In a more realistic setting for large-scale systems, as very few iterations can be allowed for each sampling time, decompositions that results from different couplings among subsystems should have a major effect on the convergence speed and thus the practical control performance obtained under early termination. Furthermore, while some distributed optimization algorithms (especially the primal–dual ones Sun and Sun, 2019; Tang and Daoutidis, 2022b) guarantees the convergence to stationary or local optima in the presence of nonconvex structures, some commonly used coordination schemes such as those based on block coordinate descent cannot provide this guarantee. In such a case, the dependence of performance on decomposition has been disclosed in the literature (Pourkargar et al., 2017, 2018; Pourkargar and Jogwar, 2021).

In this paper, we focus on the decomposition problem. Specifically, to best maintain the control performance under decomposition and reduce the computational time, the subsystems should be configured in such a way that the overall interactions across the subsystems are much weaker than inside. To this end, a network-theoretic framework is used: the structure of the dynamic system to be controlled is first represented as a network (graph) and a *community detection* procedure is applied to the network representation to generate a desirable decomposition. This approach was proposed by the authors of this paper and their coworkers in their academic research (Daoutidis et al., 2018, 2019; Tang et al., 2023), and this paper reports its implementation on our industrial process control platform with a refined community detection algorithm that is more suitable in a practical setting. By developing this decomposition feature, the subsystem configuration is assigned automatically based on the plant model, thus accelerating the APC deployment workflow and avoiding the unsystematic or erratic manual decompositions.

In this paper, we give a conceptual introduction of the design of our PACE technology and briefly review the relevant literature to elucidate the fundamental ideas underlying the decomposition strategy. We outline how community detection has been successfully implemented in PACE, and also use a challenging problem from our petrochemical processes to demonstrate the benefit of decomposition and coordination in handling large-scale systems. We believe that this exposition will, from a specific angle, contribute to better appreciation of the contemporary theory and practice of APC, of which Shell–Yokogawa's PACE technology is representative, as well as the benefits of synergy between industry and academia towards shortening the gap between theory and practice.

2. Literature review on automatic decomposition

The fact that chemical processes are multi-input–multi-output (MIMO) systems has motivated the development of input–output pairing methods in process control theory for a long time. The idea of designing multiple PID controllers for MIMO systems considering the interactions among inputs and outputs dates back to Rosenbrock (1962). The idea of relative gain array (RGA) (Bristol, 1966) has become an influential approach in process control practice and has been extended to a family of approaches, known as *interaction analysis* (McAvoy, 1983). From a robust control perspective, the interactions among the single-input–single-output (SISO) loops can be viewed as model uncertainties whose effect on closed-loop stability and performance can be quantified (Grosdidier and Morari, 1986; Yu and Fan, 1990). Hence, interaction analysis has been used to develop guidelines for synthesizing base-layer control loops (typically controllers in the PID form) on plantwide scales (Ng and Stephanopoulos, 1996; Skogestad, 2004).

In parallel to the interaction analysis research in the process control community, many works in the context of decentralized control of nonlinear systems focused on determining *graph decompositions of state–space models* (Michel et al., 1978; Vidyasagar, 1980). These studies are especially interested in the closed-loop Lyapunov stability of the entire system under separate controllers for subsystems. Graph-theoretic concepts such as connected components and acyclic directed graphs were found to be relevant (Šiljak, 1991). However, due to the structural complexities of chemical processes (with large models and recycle streams), these methods are restrictive for realistic process control applications, although in principle, the desired decomposition for plantwide optimal control should indeed be a graph-theoretic partition on the state space. Hence, for a significant period after distributed MPC was proposed (Camponogara et al., 2002), the decomposition of large-scale systems in the sense of partitioning into several MPC subsystems had not been well addressed (Christofides et al., 2013).

The emergence of network science has brought the perspective of understanding the organization of large-scale networks by investigating their macroscopic, statistical, topological features and the dynamics associated with them (Barabási, 2016). Community structure is a typical block structure existing in many biological networks (Girvan and Newman, 2002), which refers to blocks with significantly denser interconnections inside these blocks than between them. As pointed out in a number of studies (Tang and Daoutidis, 2018b; Constantino et al., 2019; Constantino and Daoutidis, 2019; Tang et al., 2019), the beneficiary role of community structures in the control of networks lies in that they lend themselves naturally to the adoption of *modular* controllers which promote the feedback sparsity (i.e., reduce complexity) while preserving the control performance. This paves the way for a systematic framework for large-scale process decomposition. That is, a network representation is first constructed for the system and then community detection is applied to generate the subsystems.

A versatile range of network representations of dynamical systems has been proposed, which flexibly capture the interactions among process variables under different characterizations. These include directed graphs (digraphs) for the interactions among manipulated inputs, states, and controlled outputs (Jogwar and Daoutidis, 2017) and bipartite graphs for input–output relations, which can be weighted by appropriately defined interaction measures (Tang and Daoutidis, 2018a; Tang et al., 2018b). In Tang et al. (2018a), network representations were proposed to directly capture the variable-constraint interactions in the optimization formulation of the MPC problem, an approach that is applicable to the decomposition of optimization problems in general (Allman et al., 2019; Mitrai and Daoutidis, 2020).

A wide range of algorithms has been proposed in the vast literature on community detection (Fortunato and Hric, 2016). The mainstream of community detection algorithms considers the problem as that of maximizing a metric called *modularity*, which is interpreted as the difference between the intra-community interconnection density and such a density in a randomized network (Newman and Girvan, 2004). According to Newman (2016), the modularity metric can be derived through a generative probabilistic modeling approach as equivalent to the log-likelihood function. As such, modularity captures the statistical significance of community structures. In the works on network decomposition for distributed control mentioned in the above paragraph, two algorithms of modularity-based community detection were mainly used. These include the *spectral algorithm* of Newman (2006) and the Louvain algorithm (also known as fast unfolding) (Blondel et al., 2008).

The main difference between the spectral algorithm and the fast unfolding algorithm is the path through which the maximum of modularity is searched for. Since the decomposition problem is combinatorial,

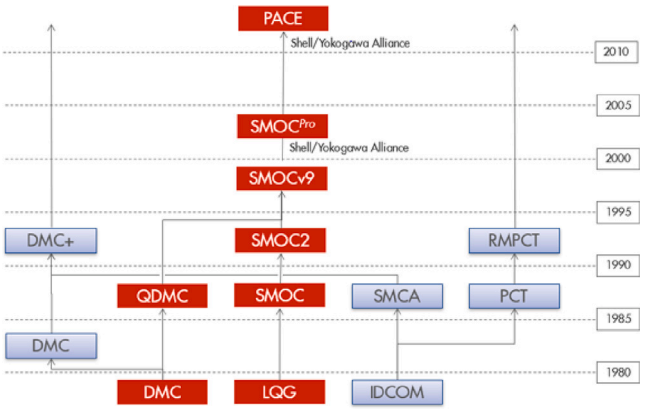


Fig. 1. History and differentiation of APC technology.

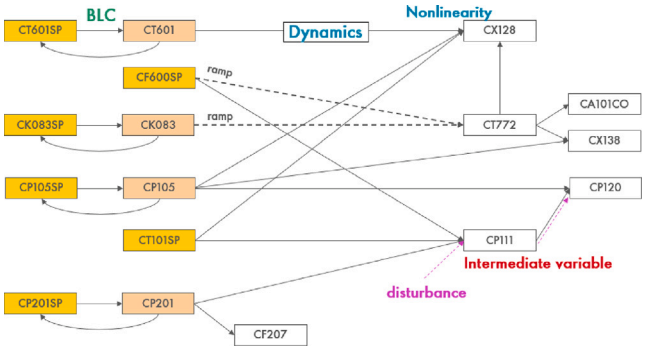


Fig. 2. Illustration of PACE model structure.

both algorithms are local and path-dependent. The former algorithm is *divisive* — it iteratively splits larger communities into smaller ones; specifically, in each iteration, a large community is tentatively cut into two (bisectioned) with maximal modularity increase, and such bisectioning is repeated until the modularity is no longer increasing. The latter algorithm is conducted in the reverse direction and is thus *agglomerative* — it is initiated from single-node clusters and in each iteration merges smaller communities into larger ones. Compared to the spectral method, the Louvain algorithm tends to reach at a higher modularity value and be more computationally efficient (Blondel et al., 2008). However, for application in distributed control, the subsystems are usually not orders-of-magnitude smaller than the entire system; it is unlikely that the number of subsystems to be used is very large, e.g., larger than 20. Therefore, the divisive algorithm results in a shorter path, while the agglomerative algorithm has a longer one where the later steps are more important than the earlier iterations. Moreover, the Louvain algorithm often generates extremely small and disconnected communities that are associated with very small incremental benefit in modularity (Traag et al., 2019). In this work, we adopt Newman's spectral algorithm and seek to refine it, as will be explained in Section 4.

The advantage of community detection-based network decomposition has been well demonstrated in the literature through simulations (Pourkargar et al., 2017, 2019). In these studies, the subsystem controllers in a distributed MPC scheme iterate their decisions according to a block coordinate descent algorithm in sequential or parallel orders. It was noted that community-based decompositions tend to result in significant decrease in computational time without sacrificing the control performance significantly. With these observations, we assert that community detection can be leveraged as a method of decomposing large-scale systems in PACE.

3. Overview of Platform for Advanced Control and Estimation (PACE)

In this section we give a high-level overview of our current technology platform, PACE, highlighting the distinctive features that differentiates it from its counterparts. Fig. 1 shows a historical roadmap of APC technology since the 1970s. The development of PACE since the 2010s at Shell is driven by the needs of improving the performance and operational acceptance, facilitating migrations and new deployments, and reducing the cost and time for software maintenance. So far, PACE has been applied to most of the refining, chemical, liquified natural gas, and gas-to-liquid plants in Shell as well as processes outside Shell through partnership with Yokogawa.

In a nutshell, PACE is an APC platform for all-in-one solution of data analysis, system identification and model building, model calibration, disturbance estimation and Kalman filtering, and static, dynamic, and economic optimization, and automatic step testing. The PACE software is divided into a "Design-Time" part, responsible for offline procedures including identification, controller design, simulations, and a "Run-Time" part for online controller configuration and tuning, step tests, and performance reporting (Carrette, 2020).

3.1. Modeling and step testing

PACE enables the modeling of rich cause-and-effect structures. As illustrated in Fig. 2, the skeleton of a PACE model is constituted of dynamics (transfer functions) from manipulated variables (MVs) and disturbance variables (DVs) to intermediate variables (IVs) to process output variables (POVs). The involvement of IVs allows the modeling of complex systems convoluted from simpler transfer functions, easier maintenance, and the feedforward of non-output measurements as calibration contributions. It also facilitates the incorporation of base-layer control (BLC) models, ranging from simple BLC loops, saturated, coupled, to cascaded ones, in a highly flexible manner. Nonlinear blocks can also be specified for POVs based on equations.

The step test is a crucial yet expensive prerequisite procedure of collecting the necessary data for model identification. In PACE, the step test becomes automated with an internal package integrated with control, thus simplifying the transition between control and automated testing. The exciting signals for the auto-step are designed such that the information content contained in data is optimized. During the step test, constraints are well managed, which is combined with the planning of subsequent experiments.

3.2. Controller tuning and calibration

The way that static and dynamic optimization problems are handled in PACE improves both the operational flexibility and accuracy. First, in static optimization, multiple economic functions (EFs) specified by linear or nonlinear equations, together with controlled variable (CV) specifications, can be user-defined with a hierarchy of priorities. Second, the dynamic optimization formulation is generated from the controller tuning parameters. Instead of tuning the weighting matrices (Q, R) as in usual MPC, the PACE users specify speeds of MV response, CV response and dynamic tracking of the EFs. Such a tuning strategy allows more consistent and accurate responses, management of overshooting, and also adaptation to activation/inactivation of CVs. Moreover, a state-of-the-art commercial nonlinear optimization solver replaces traditional linear programming (LP) solvers for handling nonlinear processes and obtaining the rigorous optimal solution.

To achieve offset-free control (Muske and Badgwell, 2002; Pannocchia and Rawlings, 2003), the optimization formulations involve a calibration mechanism, which accounts for and aims at eliminating the discrepancies between model prediction and actual POV values.

PACE calibration is a combination of (i) event detection for detecting fast significant disturbances and (ii) Kalman filter for handling usual white noise sources (with the filtering speed tuned by the user). The calibration is furthermore de-tuned for robustness, by assuming uncertainty factors on the transfer function models. In its unmeasured disturbance modeling, PACE incorporates both input and output disturbances, whose shape parameters are tunable. Such flexible designs, along with the use of IVs in the model structure, allows the calibration algorithm to better capture the disturbances than the classical bias estimation and update scheme (e.g., Cutler and Ramaker (1979)).

3.3. Subsystems in PACE

At the present time, equipped with the decomposition and coordination strategies, PACE is developing its capability of handling large-scale challenging problems. In PACE, the configuration of subsystems can be either directly and manually user-specified (i.e., the user assigns all MVs and POVs or part of them into several subsystems), or preferably determined automatically by community detection. With the subsystem configuration specified, the process variables and model blocks are contained in the subsystems, and hence the optimization variables, constraints, and objectives in the dynamic optimization problems of the subsystems are automatically formulated. The distributed MPC essentially applies an iterative algorithm where in each iteration, the subsystem problems are solved through parallel computing. For online implementation, the computational time allowed for distributed optimization computation is highly restricted. In practice, we are typically limited to a single or a few iteration in every sampling period. The solutions obtained from such early termination should be tested before commissioning, and ad hoc measures can be taken to ensure that the closed-loop dynamic behavior is satisfactory.

The iterative algorithm that we currently use involves the following treatment:

- For the dynamic optimization problem of each subsystem, the MV and CV weights are tuned based on the subsystem model alone. In addition, quadratic terms associated with the *shared variables* (i.e., the variables that affect other subsystems) are added to the subsystem's objective function, and such quadratic terms are tuned as if they are CVs.
- After each iteration, the solutions associated with the predicted trajectories of the shared variables are fed forward to their downstream subsystems (i.e., those that they affect), as if such shared variables were DVs. In this way, the optimization problems are restricted to the variables inside subsystems.
- The convex combination scheme of Stewart et al. (2011) is used, i.e., if $\hat{u}_i^*(t)^{[k+1]}$ is the optimal solution for MV u_i at predicted time t in iteration $k+1$, only a fraction $\beta_i \in [0,1]$ of the update will be actually taken:

$$\hat{u}_i(t)^{[k+1]} = \hat{u}_i(t)^{[k]} + \beta_i \left(\hat{u}_i^*(t)^{[k+1]} - \hat{u}_i(t)^{[k]}\right). \tag{1}$$

  Here, the coefficients β_i, dependent only on the MV indices, are rationally chosen based on a Hessian approximation procedure.

In the literature, relevant progress has been made for real-time iterations in centralized MPC (Diehl et al., 2005; Yang and Biegler, 2013) and acceleration techniques in distributed optimization (Tang and Daoutidis, 2022b, 2021). For PACE, we are developing a new scheme of coordinating subsystem controllers that aims to account for the effect of each controller's actions on other subsystems. This helps to use few iterations to reach a close-to-centralized coordinated performance under automatic decomposition. It is beyond the scope of this paper to disclose its details. We believe that the development of more efficient and real-time implementable distributed optimization algorithms with guaranteed performance is an important direction of future research.

4. Community detection in PACE

In the above sections, we have reviewed the literature about decomposing large-scale systems through community detection in networks and introduced the main technical features of PACE. Now we present how automatic decomposition is realized in PACE.

4.1. Network representation

As described previously, the main body of the model structure in PACE comprises model blocks (transfer functions) from MVs, IVs, and POVs. This naturally allows the construction of a *directed network* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the set of nodes $\mathcal{V} = \{1, 2, \dots, n\}$ represent the variables, and each directed edge (i, j) in the edge set $\mathcal{E}$ corresponds to the model block from variable i to variable j, if such a model block exists. The topology of the directed network can be represented by a sparse *adjacency matrix* A, where its (i, j)th entry $a_{ij} = 1$ if $(i, j) \in \mathcal{E}$ and $a_{ij} = 0$ if $(i, j) \notin \mathcal{E}$. In our current implementation, we do not assign the edges with weights due to the following reasons.

1. During the system identification procedure, the variables should have been properly re-scaled, and the weak interactions are pruned. In other words, the system identification procedure is responsible for scaling and discriminating interaction strengths.
2. The decomposition should be such that the subsystems have minimal *number* of interactions among them rather than total *weights*, since the number of coupling interactions may require more iterative coordination and thus affect the computational performance more significantly.
3. There does not exist a rigorous and clearly defined way of defining edge weights that guarantees any property of the resulting distributed MPC, although a few reasonable heuristic metrics exist, e.g., in Jogwar (2019), Pourkargar and Jogwar (2021) and Wang et al. (2023). In PACE, we tested a few different options of edge weighting, none of which showed a clear advantage over undirected network. We are aware that such a conclusion may vary with different operating conditions or system models and controller tuning.

We also perform some pre-processing steps on the network representation. These procedures are necessary to ensure that the complexity (i.e., the internal heterogeneity in the nature of variables and their interactions) of the PACE model is resolved before the decomposition and that the result can be better understood and accepted by the user.

- The variables that do not participate in the dynamic optimization, including disturbance variables and inactivated MVs, are removed from the network and unassigned in the end. If such removal results in isolated nodes (which does not connect to any other node), then the isolated nodes are not considered for community detection, but only assigned to a separate subsystem afterwards.
- For each BLC loop, the variables involved should be assigned to the same subsystem. Hence, these BLC variables are considered as indivisible groups and first agglomerated as a single node. The same is carried out for the variables involved in each static nonlinear transformation.

4.2. Modularity and resolution parameter tuning

The community detection in a directed network aims at maximizing the *modularity* (Leicht and Newman, 2008), which is a function of all possible partitions $g = (g_1, \dots, g_n)$:

$$Q(g) = \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{1}{m}\left(a_{ij} - \gamma \frac{k_i^+ k_j^-}{m}\right)\delta_{g_i g_j}, \tag{2}$$

where k_i^+ is the out-degree of node i, k_j^- is the in-degree of node j, and m is the total number of edges:

$$k_i^+ = \sum_{j=1}^{n} a_{ij}, \quad k_j^- = \sum_{i=1}^{n} a_{ij}, \quad m = \sum_{i=1}^{n} k_i^+ = \sum_{j=1}^{n} k_j^-. \tag{3}$$

g_i is the index of the community to which node i belongs, and δ is Kronecker's delta, i.e., $\delta_{g_i g_j} = 1$ if nodes i and j are in the same community and 0 otherwise. Thus, $k_i^+ k_j^-/m$ is regarded as the expected number of edges between nodes i and j in a randomized network and therefore a "standard threshold" whose difference with a_{ij} is the extent to which these two nodes are preferred to be in the same community. The parameter $\gamma > 0$ here, called the *resolution parameter*, offers a tuning of this threshold (Reichardt and Bornholdt, 2006). When γ is increased, the community detection tends to find a larger number of smaller communities, while smaller γ promotes a coarser decomposition.

For the user's convenience, we allow the user to simply specify the desired number of communities K and an algorithm is used to adaptively find the resolution γ such that the resulting decomposition is into K subsystems.

- If it is known that at γ_1 and γ_2, the maximization of Q leads to $K_1 < K$ and $K_2 > K$ communities, respectively, then assuming that $\ln K$ and $\ln \gamma$ have a linear relation approximately,[1] we update γ by

$$\ln \gamma = \frac{\ln K_2 - \ln K}{\ln K_2 - \ln K_1} \ln \gamma_1 + \frac{\ln K - \ln K_1}{\ln K_2 - \ln K_1} \ln \gamma_2. \tag{4}$$

- At the first iteration, $\gamma = 1$ is used. When either γ_1 and γ_2 is not known (without loss of generality, say that only γ_1 is known, giving a $K_1 < K$), then by assuming that $K \propto \gamma$, we update the resolution parameter by

$$\gamma = \frac{K}{K_1} \gamma_1. \tag{5}$$

- After the update, if under γ the number of communities is exactly K, then we terminate the iterations. Otherwise we update γ_1 and K_1 or γ_2 and K_2 according to the rules above.

Empirically, we found that the above rules allow us to find the correct resolution parameter for a given K within 10 iterations.

4.3. Spectral algorithm for recursive bisectioning

Under a given γ, the maximization of modularity Q follows a recursive bisectioning procedure. Defining $c_{ij} = a_{ij} - \gamma k_i^+ k_j^-/m$, as pointed out in Newman (2006), when a community S is partitioned into two sub-communities S_+ and S_-, the resulting modularity increase is

$$\Delta Q(s) = \frac{1}{m} \sum_{i \in S} \sum_{j \in S} \left(s_i s_j - 1\right) c_{ij} = \frac{1}{m} \left(s^\top C_S s - e^\top C_S e\right), \tag{6}$$

where $s = (s_i)_{i \in S}$, $s_i = +1$ if $i \in S_+$ and -1 if $i \in S_-$, $C_S = [c_{ij}]_{i,j \in S}$, and e is a vector (of length $|S|$) whose all elements are equal to 1.

To maximize $\Delta Q(s)$ with respect to $s \in \{-1, 1\}^{|S|}$, the following techniques are used.

- An approximate solution $s = \text{sign}(v_1(C'_S))$ is taken first, in which $C'_S = (C_S + C_S^\top)/2$, $v_1(\cdot)$ refers to the unit vector associated with the largest eigenvalue of the matrix, and $\text{sign}(\cdot)$ is element-wise sign function.
- The vector s is further *fine-tuned* by tentatively flipping the sign of each component, and the flipping with the maximum increase in ΔQ is accepted each time.

In the end, if the maximized $\Delta Q(s)$ is above a threshold value $\alpha = 10^{-3}$, then the bisectioning is accepted. Such a threshold value prevents the production of extremely small communities.[2] Also, to guarantee the numerical stability of fine-tuning, we require that for the sign flipping to be accepted, its resulting modularity increase must be at least $\alpha/10$ and the total number of such flippings in each bisectioning should not exceed the number of nodes in S.

4.4. Connectedness restoration and load balancing

It is possible that after the spectral method, the communities found are not connected inside themselves, in which case the subsystems cannot be considered as physically coherent portions of the process. Also, the sizes of communities may differ significantly, which is undesirable from the perspective of parallel computing in distributed MPC. Therefore, we carry out two major post-processing steps after modularity maximization.

First, in every community detected, a depth-first search (DFS) is performed to characterize all its *connected components* (i.e., subgraphs in which every pair of nodes are connected by at least an undirected path). Except for the largest connected component, which will preserve the identity of the community, every remaining connected component is moved into another community. This destination community is chosen as the one with which the connected components have most connections, and in the case that the connected component has no connection with any other community, i.e., is isolated, a new community is created for the isolated component. Such adjustment steps are repeated until no community has more than 1 connected components. An illustration of this connectedness restoration step is given in Fig. 3(a).

Load balancing is then performed by recursively merging the smallest community into a larger one.

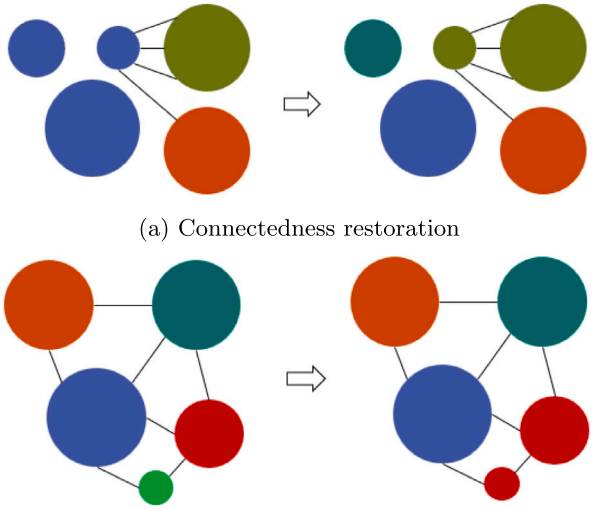
- Suppose that before balancing, the largest community has n_1 nodes. Then we consider the "effective" number of communities as $K_e = \lfloor n/n_1 \rfloor$ and merging should be done for the communities smaller than the K_eth one.
- For each small community to be merged, we look for a destination community whose size, when added to the size of this small community, is closest to n/K_e.
- We also require the two communities to merge to be connected. If a community does not connect to any other one, then this criterion is not applied.

This procedure is illustrated in Fig. 3(b). Note that the number of communities K under the γ should be comprehended as the K_e here after load balancing.

The post-processing steps have a minor effect on modularity, typically resulting in a small decrement within 0.02. This is in accordance with the observation in network science studies that by perturbing the community structures on the macroscopic level, there are often degenerate community configurations that correspond to similar modularity values (Good et al., 2010). The study of Riolo (2020) revealed that when such degeneracy appears, the network can have essential, lower-scale community structures that are not affected by high-level rearrangements, although such true communities are not explicitly and directly exposed by modularity maximization. We implicitly accept such a postulate as the modularity-based community detection approach is adopted. A deeper analysis on the consistency of community landscape and its relation with the network properties can be found in Peixoto (2021), and robust community detection has been the theme of some more recent studies, e.g., Silva et al. (2022).

[1] According to the statistical interpretation of Newman (2006), the resolution γ should be chosen as $\gamma = (\omega_0 - \omega_1)/(\ln \omega_0 - \ln \omega_1)$, where the "propensity parameters" ω_0 and ω_1 are such that if $g_i = g_j$, the expectation of a_{ij} is $\omega_0 k_i^+ k_j^-/m$, and if $g_i \neq g_j$, the expectation of a_{ij} is $\omega_1 k_i^+ k_j^-/m$. Suppose that we have K communities, and the fraction of edges across communities is $\varepsilon \ll 1$, then we should have $\omega_0 + (K-1)\omega_1 = K$ and $(1-\varepsilon)/\omega_0 + \varepsilon/\omega_1 = 1$, which results in $\gamma \approx K/\ln(1/\varepsilon)$. Therefore, γ is proportional to K, i.e., $\ln \gamma$ and $\ln K$ has a linear relation.

[2] Consider a large network with n nodes in which there are $\alpha \cdot n$ nodes ($\alpha \ll 1$) not connected to the rest of the network. Assigning this group of nodes, denoted as S, into a separate community leads to $\Delta Q = \sum_{i \in S} \sum_{j \notin S} k_i^+ k_j^-/m^2 \approx m_S/m$. Here m_S is the number of edges inside S. If the edge distributions in and out of S are uniform, then $\Delta Q \approx m_S/m \approx \alpha$.



(a) Connectedness restoration

(b) Load balancing

Fig. 3. Illustration of postprocessing procedures.

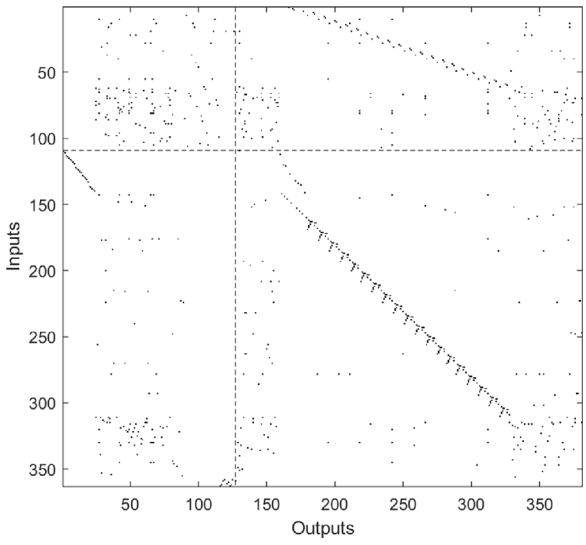
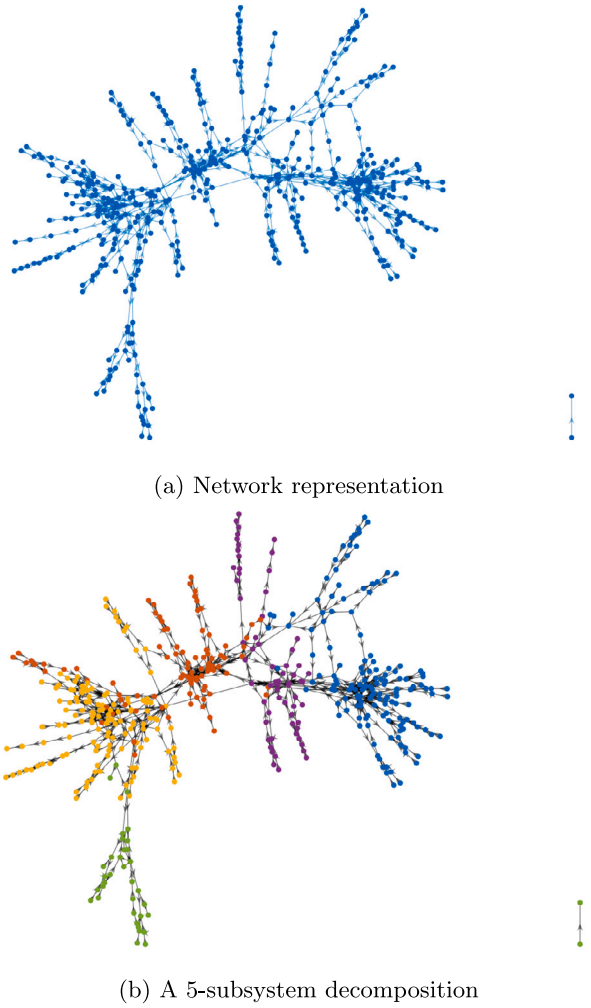


Fig. 4. Model structure of the crude distillation column process. (A black pixel stands for the presence of a model block. The dashed lines separate non-IVs (upper among inputs, left among outputs) and IVs (lower among inputs, right among outputs).

5. Case study

For the purpose of illustration, we consider a crude distillation process for a refinery, whose model contains 363 inputs, 381 outputs (among which there are 246 intermediate variables that are both inputs and outputs), and 801 model blocks, as visualized in Fig. 4. A full formulation of its dynamic optimization problem contains 875 972 rows (constraints) and 828 016 columns (variables), with 2 430 226 non-zero relations between the variables and constraints. To our best knowledge, there has not been any work in the literature that addresses distributed MPC on a comparable plantwide scale.

As a result, the state-of-the-art optimization solver used in PACE takes 35.4 s to solve a single centralized dynamic optimization problem on average, compared to the sampling time of 60 s. Such a controller



(a) Network representation

(b) A 5-subsystem decomposition

Fig. 5. Automatic decomposition of a crude distillation process. (Different colors correspond to communities.)

implemented for online operations has been well known to suffer from severe delays and also frequent shut-off due to computational timeout.[3]

The directed network of variables that represents this process system is shown in Fig. 5(a), with 476 nodes and 787 edges. An intuitive glance at the network topology suggests that there exists community structure in the process, hence decomposing such a system is desirable. By examining the computational time and control performance under different numbers of communities K, we empirically set $K = 5$, which results in a good trade-off. The resulting decomposition into 5 communities is shown in Fig. 5(b). The edges lying across communities is 31, which is 3.9% of the total number of edges, and the number of inputs impacting outputs in other subsystems is 29, which is 8.0% of the total number of inputs. These indices demonstrate that the subsystems are indeed weakly coupled.

After the decomposition by community detection, a simulation is run for the closed-loop system. The simulated scenario considers disturbances happening on several MVs and POVs. The trajectories under centralized MPC and under distributed MPC exhibit highly similar behaviors, which is omitted for brevity here. Before using automatic

[3] The average computational time mentioned here was collected under the simulation of a typical normal operating condition. Occasionally, due to the parameter changes in the dynamic optimization problem, the computational time can reach over 70 s.

Table 1
Outcomes of automatic decomposition on benchmark processes.

Process	Crude distillation	GTL	Hydrocracking
Number of MVs	117	68	25
Number of CVs	381	412	117
Computational time for centralized MPC (s)	35.4	16.0	10.0
Subsystems	5	4	4
Computational time with decomposition (s)	6.1	3.9	3.3
Acceleration factor	5.8	4.1	3.0

decomposition, the site engineers manually assigned a 7-subsystem decomposition, which takes over 20 s on average for each dynamic optimization, as the subsystems are not well balanced. The simulation trajectories under the manual decomposition also deviates from those under centralized control. We are therefore inclined to believe that merely applying domain knowledge is often not sufficient to determine a desirable decomposition within a complex plant. In contrast, with afore-mentioned automatic decomposition, the average computational time for solving the dynamic optimization problem at each sampling time is now reduced down to 6.1 s,[4] with an acceleration factor of 5.8.

Remarkably, such significant improvement in the computational performance of dynamic optimization is also observed in several other benchmark processes of Shell, including a gas-to-liquid (GTL) process and a hydrocracking process. Their model information and computational time results are shown in Table 1.

6. Conclusions

In this paper, we focus on the idea of *automatic decomposition*, which is necessary for structured, systematic solution of dynamic optimization problems arising in the MPC of large-scale systems. After a literature review on the evolution of relevant academic research, we present the successful implementation of an automatic decomposition method in the Shell–Yokogawa's new-generation APC platform — PACE, and show its advantages through an application to a real-world large-scale process. We thus demonstrate how this fundamental idea of decomposition, originating in the early ages of APC and enhanced with very recent advances in academic research, is shaping and empowering modern, leading process control technology.

CRediT authorship contribution statement

Wentao Tang: Conceptualization, Investigation, Software, Writing – original draft, Writing – review & editing, Validation, Visualization. **Pierre Carrette:** Project administration, Software, Writing – review & editing. **Yongsong Cai:** Conceptualization, Investigation, Software. **John M. Williamson:** Funding acquisition, Supervision. **Prodromos Daoutidis:** Conceptualization, Funding acquisition, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

[4] The computational time with the decomposition is the *actual* total time costed by the CPU for solving the dynamic optimization problem using multi-thread parallel computing, including the overhead time.

Acknowledgment

This work was supported by NSF-CBET Award 1926303 and the Shell Global Solutions (U.S.) Inc.

References

Allman, A., Tang, W., Daoutidis, P., 2019. DeCODe: a community-based algorithm for generating high-quality decompositions of optimization problems. Optim. Eng. 20 (4), 1067–1084.
Amrit, R., Canney, W., Carrette, P., Linn, R., Martinez, A., Singh, A., Skrovanek, T., Valiquette, J., Williamson, J., Zhou, J., 2015. Platform for Advanced Control and Estimation (PACE): Shell's and Yokogawa's next generation advanced process control technology. IFAC-PapersOnLine 48 (8), 1–5.
Baldea, M., Daoutidis, P., 2012. Dynamics and Nonlinear Control of Integrated Process Systems. Cambridge University Press.
Barabási, A.-L., 2016. Network Science. Cambridge University Press.
Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E., 2008. Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. 2008 (10), P10008.
Bristol, E., 1966. On a new measure of interaction for multivariable process control. IEEE Trans. Automat. Control 11 (1), 133–134.
Camponogara, E., Jia, D., Krogh, B.H., Talukdar, S., 2002. Distributed model predictive control. IEEE Control Syst. Mag. 22 (1), 44–52.
Carrette, P., 2020. APC technology in Shell: Levaraging technical advancements for increased benefits. In: Texas-Wisconsin-California Control Consortium.
Chen, S.W., Wang, T., Atanasov, N., Kumar, V., Morari, M., 2022. Large scale model predictive control with neural networks and primal active sets. Automatica 135, 109947.
Christofides, P.D., Scattolini, R., Muñoz de la Peña, D., Liu, J., 2013. Distributed model predictive control: A tutorial review and future research directions. Comput. Chem. Eng. 51, 21–41.
Constantino, P.H., Daoutidis, P., 2019. A control perspective on the evolution of biological modularity. IFAC-PapersOnLine 52 (11), 172–177.
Constantino, P.H., Tang, W., Daoutidis, P., 2019. Topology effects on sparse control of complex networks with Laplacian dynamics. Sci. Rep. 9, 9034.
Cott, B.J., 2007. Unit-wide model predictive control with SMOCPro. In: CSChE Annual Conference. In: D.G. Fisher Award Keynote Lecture.
Cutler, C.R., Ramaker, B.L., 1979. DMC – a computer control algorithm. In: AIChE Annual Meeting.
Daoutidis, P., Tang, W., Allman, A., 2019. Decomposition of control and optimization problems by network structure: concepts, methods and inspirations from biology. AIChE J. 65 (10), e16708.
Daoutidis, P., Tang, W., Jogwar, S.S., 2018. Decomposing complex plants for distributed control: perspectives from network theory. Comput. Chem. Eng. 114, 43–51.
Diehl, M., Bock, H.G., Schlöder, J.P., 2005. A real-time iteration scheme for nonlinear optimization in optimal feedback control. SIAM J. Control Optim. 43 (5), 1714–1736.
Fortunato, S., Hric, D., 2016. Community detection in networks: A user guide. Phys. Rep. 659, 1–44.
Garcia, C.E., Morshedi, A., 1986. Quadratic programming solution of dynamic matrix control (QDMC). Chem. Eng. Commun. 46 (1–3), 73–87.
Girvan, M., Newman, M.E.J., 2002. Community structure in social and biological networks. Proc. Natl. Acad. Sci. USA 99 (12), 7821–7826.
Good, B.H., De Montjoye, Y.-A., Clauset, A., 2010. Performance of modularity maximization in practical contexts. Phys. Rev. E 81 (4), 046106.
Grosdidier, P., Morari, M., 1986. Interaction measures for systems under decentralized control. Automatica 22 (3), 309–319.
Jogwar, S.S., 2019. Distributed control architecture synthesis for integrated process networks through maximization of strength of input–output impact. J. Process Control 83, 77–87.
Jogwar, S.S., Daoutidis, P., 2017. Community-based synthesis of distributed control architectures for integrated process networks. Chem. Eng. Sci. 172, 434–443.
Katz, J., Pappas, I., Avraamidou, S., Pistikopoulos, E.N., 2020. The integration of explicit MPC and ReLU based neural networks. IFAC-PapersOnLine 53 (2), 11350–11355.
Kumar, P., Rawlings, J.B., Wright, S.J., 2021. Industrial, large-scale model predictive control with structured neural networks. Comput. Chem. Eng. 150, 107291.

Leicht, E.A., Newman, M.E.J., 2008. Community structure in directed networks. Phys. Rev. Lett. 100 (11), 118703.
Marquis, P., Broustail, J.P., 1988. SMOC, a bridge between state space and model predictive controllers: application to the automation of a hydrotreating unit. IFAC Proc. Vol. 21 (4), 37–45.
McAvoy, T.J., 1983. Interaction Analysis: Principles and Applications. ISA.
Michel, A., Miller, R., Tang, W., 1978. Lyapunov stability of interconnected systems: Decomposition into strongly connected subsystems. IEEE Trans. Circuits Syst. 25 (9), 799–809.
Mitrai, I., Daoutidis, P., 2020. Decomposition of integrated scheduling and dynamic optimization problems using community detection. J. Process Control 90, 63–74.
Morari, M., Arkun, Y., Stephanopoulos, G., 1980. Studies in the synthesis of control structures for chemical processes: Part I: Formulation of the problem. Process decomposition and the classification of the control tasks. Analysis of the optimizing control structures. AIChE J. 26 (2), 220–232.
Muske, K.R., Badgwell, T.A., 2002. Disturbance modeling for offset-free linear model predictive control. J. Process Control 12 (5), 617–632.
Newman, M.E.J., 2006. Modularity and community structure in networks. Proc. Natl. Acad. Sci. USA 103 (23), 8577–8582.
Newman, M.E.J., 2016. Equivalence between modularity optimization and maximum likelihood methods for community detection. Phys. Rev. E 94 (5), 052315.
Newman, M.E.J., Girvan, M., 2004. Finding and evaluating community structure in networks. Phys. Rev. E 69 (2), 026113.
Ng, C.S., Stephanopoulos, G., 1996. Synthesis of control systems for chemical plants. Comput. Chem. Eng. 20, S999–S1004.
Pannocchia, G., Rawlings, J.B., 2003. Disturbance models for offset-free model-predictive control. AIChE J. 49 (2), 426–437.
Peixoto, T.P., 2021. Revealing consensus and dissensus between network partitions. Phys. Rev. X 11 (2), 021003.
Pourkargar, D.B., Almansoori, A., Daoutidis, P., 2017. Impact of decomposition on distributed model predictive control: A process network case study. Ind. Eng. Chem. Res. 56 (34), 9606–9616.
Pourkargar, D.B., Almansoori, A., Daoutidis, P., 2018. Comprehensive study of decomposition effects on distributed output tracking of an integrated process over a wide operating range. Chem. Eng. Res. Des. 134, 553–563.
Pourkargar, D.B., Jogwar, S.S., 2021. Distributed model predictive control of integrated process networks: Optimal decomposition for varying operating point. In: Am. Control Conf. (ACC). IEEE, pp. 801–807.
Pourkargar, D.B., Moharir, M., Almansoori, A., Daoutidis, P., 2019. Distributed estimation and nonlinear model predictive control using community detection. Ind. Eng. Chem. Res. 58 (30), 13495–13507.
Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control Eng. Pract. 11 (7), 733–764.
Rawlings, J.B., Mayne, D.Q., Diehl, M.M., 2017. Model Predictive Control: Theory, Computation, and Design, 2nd ed. Nob Hill Publishing.
Reichardt, J., Bornholdt, S., 2006. Statistical mechanics of community detection. Phys. Rev. E 74 (1), 016110.
Riolo, M.A., 2020. Consistency of community structure in complex networks. Phys. Rev. E 101 (5), 052306.
Rosenbrock, H.H., 1962. Distinctive problems in process control. Chem. Eng. Prog. 58 (9), 43–50.
Scattolini, R., 2009. Architectures for distributed and hierarchical model predictive control – a review. J. Process Control 19 (5), 723–731.
Šiljak, D.D., 1991. Decentralized Control of Complex Systems. Academic Press.
Silva, F.N., Albeshri, A., Thayananthan, V., Alhalabi, W., Fortunato, S., 2022. Robustness modularity in complex networks. Phys. Rev. E 105 (5), 054308.
Skogestad, S., 2004. Control structure design for complete chemical plants. Comput. Chem. Eng. 28 (1–2), 219–234.
Stewart, B.T., Wright, S.J., Rawlings, J.B., 2011. Cooperative distributed model predictive control for nonlinear systems. J. Process Control 21 (5), 698–704.
Sun, K., Sun, X.A., 2019. A two-level distributed algorithm for general constrained non-convex optimization with global convergence. arXiv preprint arXiv:1902.07654.
Tang, W., Allman, A., Mitrai, I., Daoutidis, P., 2023. Resolving large-scale control and optimization through network structure analysis and decomposition: A tutorial review. In: Am. Control Conf. (ACC). IEEE, pp. 3113–3129.
Tang, W., Allman, A., Pourkargar, D.B., Daoutidis, P., 2018a. Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection. Comput. Chem. Eng. 111, 43–54.
Tang, W., Babaei Pourkargar, D., Daoutidis, P., 2018b. Relative time-averaged gain array (RTAGA) for distributed control-oriented network decomposition. AIChE J. 64 (5), 1682–1690.
Tang, W., Constantino, P.H., Daoutidis, P., 2019. Optimal sparse network topology under sparse control in Laplacian networks. IFAC-PapersOnLine 52 (20), 273–278.
Tang, W., Daoutidis, P., 2018a. Network decomposition for distributed control through community detection in input–output bipartite graphs. J. Process Control 64, 7–14.
Tang, W., Daoutidis, P., 2018b. The role of community structures in sparse feedback control. In: Am. Control Conf. (ACC). IEEE, pp. 1790–1795.
Tang, W., Daoutidis, P., 2021. Coordinating distributed MPC efficiently on a plantwide scale: The Lyapunov envelope algorithm. Comput. Chem. Eng. 155, 107532.
Tang, W., Daoutidis, P., 2022a. Data-driven control: Overview and perspectives. In: Am. Control Conf. (ACC). IEEE, pp. 1048–1064.
Tang, W., Daoutidis, P., 2022b. Fast and stable nonconvex constrained distributed optimization: the ELLADA algorithm. Optim. Eng. 23, 259–301.
Traag, V.A., Waltman, L., Van Eck, N.J., 2019. From Louvain to Leiden: guaranteeing well-connected communities. Sci. Rep. 9 (1), 5233.
Vidyasagar, M., 1980. Decomposition techniques for large-scale systems with nonadditive interactions: Stability and stabilizability. IEEE Trans. Automat. Control 25 (4), 773–779.
Wang, J., Song, C., Zhao, J., Mo, Z., Xu, Z., 2023. Distributed model predictive control-oriented network decomposition based on full dynamic response. AIChE J. 69 (1), e17951.
Yang, X., Biegler, L.T., 2013. Advanced-multi-step nonlinear model predictive control. J. Process Control 23 (8), 1116–1128.
Yu, C.-C., Fan, M.K.H., 1990. Decentralized integral controllability and D-stability. Chem. Eng. Sci. 45 (11), 3299–3309.