Distributed and Localized Model Predictive Control via System Level Synthesis

Carmen Amo Alonso[1] and Nikolai Matni[2]

***Abstract*— We present the Distributed and Localized Model Predictive Control (DLMPC) algorithm for large-scale structured linear systems, wherein only local state and model information needs to be exchanged between subsystems for the computation and implementation of control actions. We use the System Level Synthesis (SLS) framework to reformulate the MPC problem as an optimization problem over closed loop system responses, and show that this allows us to naturally impose localized communication constraints between sub-controllers, such that only local state and system model information needs to be exchanged for both computation and implementation of closed loop MPC control policies. In particular, we show that the structure of the resulting optimization problem can be exploited to develop an Alternating Direction Method of Multipliers (ADMM) based algorithm that allows for distributed and localized computation of control decisions. Moreover, our approach can accommodate constraints and objective functions that couple the behavior of different subsystems, so long as the coupled systems are able to communicate directly with each other, allowing for a broader class of MPC problems to be solved via distributed optimization. We conclude with numerical simulations to demonstrate the usefulness of our method, and in particular, we demonstrate that the computational complexity of the subproblems solved by each subsystem in DLMPC is independent of the size of the global system.**

I. INTRODUCTION

Model Predictive Control (MPC) has been shown to provide broadly applicable solutions for many industrial applications. However, the recent need to control increasingly large-scale distributed systems - such as power networks, intelligent transportation systems, and communication networks - poses a challenge. Large-scale distributed systems are often impossible to control with a centralized controller, and moreover, even when such a centralized controller is implementable, the high computational demand of MPC renders it impractical. Thus, efforts have been made to develop *distributed* MPC algorithms, wherein each sub-controller solves a local optimization problem and coordinates appropriately with other sub-controllers.

The degree of coordination allowed between sub-controllers plays a key role in determining the tractability of distributed optimal control problems [1], and this is equally true in the context of distributed MPC. As noted in [2], the kind of coordination that can be achieved is largely dependent on the topology of the communication network across which sub-controllers can exchange information. If a network is strongly connected and allows for global communication then every subsystem in the network (eventually) receives information about every other subsystem; conversely, if only local communication is allowed, each subsystem can only receive information from those sub-controllers within its local set of neighbors. Distributed MPC algorithms relying on a global communication scheme have proved to be very successful when applied to systems of moderate size ([3], [4]). However, this approach does not scale well with the size of the full system, as global information sharing is required amongst all sub-controllers, limiting its applicability.

In response to this, Distributed MPC schemes requiring only local exchange of information between sub-controllers have become prominent ([5], [6], [7], [8]). In particular, [6] takes advantage of the sparsity of the system and presents a distributed MPC algorithm that scales gracefully with the size of the network. However, the presented algorithm does not take into account how information propagates in the network - the so called information structure [9] - and only focuses on the influence from immediate neighbors. Moreover, this work relies on the assumption that a structured localized controller exists, and in order for such a controller to be implementable, a relaxation is necessary, which leads to conservatism in the solution. Another approach was taken in [7], where the authors consider the information structure of the network through an inflow-outflow approach. With only local communication they are able to converge to a globally optimal solution. However, sparsity is not exploited and in order to reduce the computational burden for large networks, once again, an approximation is required.

It therefore remains an open problem as to how to design a distributed MPC algorithm for large-scale systems that takes into account the information exchange constraints of the system and that allows for a distributed implementation and computation of the control law without introducing any approximations or conservatism. This paper closes this gap, and shows that by maximally exploiting the structure of the underlying distributed system, we can define and implement a distributed optimization algorithm that converges to globally optimal solutions, and for which the computational complexity of the sub-problems solved at each subsystem is constant, i.e., $O(1)$, relative to the size of the global system. In particular, we present the Distributed Localized MPC (DLMPC) algorithm for linear time-variant discrete time systems, which requires only local communication of state information and system models between subsystems. By reformulating the MPC problem in the System Level

[1]C. Amo Alonso is a graduate student in the Computing and Mathematical Sciences Department at California Institute of Technology, Pasadena, CA. camoalon@caltech.edu

[2]N. Matni is an Assistant Professor with the Department of Electrical and Systems Engineering at the University of Pennsylvania, Philadelphia, PA. nmatni@seas.upenn.edu

Synthesis (SLS) framework [10], [11], [12], we are able to naturally take into account the information structure of the network by imposing localized constraints on the system responses, and consequently, the communication structure of the resulting controller. In particular, we are able to verify - rather than assume - that a controller with localized implementation exists. Furthermore, by exploiting the sparsity the resulting closed loop system and the separability properties of often used objective functions and constraints (e.g., quadratic costs subject to polytopic constraints), we are able to distribute the computation via the Alternating Direction Method of Multipliers (ADMM), thus allowing for the online computation of closed loop MPC control policies to be done in a distributed and localized manner. Hence, in the resulting implementation each sub-controller solves a low-dimensional optimization problem defined over its local neighborhood, requiring only local communication of state and model information. We show that no conservatism or approximations are introduced via our formulation, and that under suitable (and standard) regularity assumptions, the algorithm converges to the globally optimal solution. Furthermore, we show that generic convex constraints and objective functions that couple states/inputs of neighboring subsystems can be computed using a distributed implementation via a consensus-like algorithm. To the best of our knowledge, no work before has dealt with such a general class of objective functions and constraints. Through extensive simulations, we demonstrate that by exploiting system structure and localizing communication, the computational complexity of the sub-problems solved at each sub-controller scales as $O(1)$ relative to the full size of the system.

The remainder of this paper is organized as follows. In Section II we present the problem formulation, and in Section III, we introduce the SLS framework and show how to recast the standard MPC problem as an optimization problem over system responses. In Section IV we define and analyze the DLMPC algorithm: in the interest of clarity, we first present our approach for the simplified setting of uncoupled constraints and objectives, and then build upon this simplified setting to present the general approach applicable to convex constraints and objectives that introduce local coupling between subsystems. Section V provides a numerical study to assess the efficacy of the proposed algorithm. In Section VI we end with conclusions and a discussion of directions for future work.

NOTATION

Lower-case and upper-case Latin and Greek letters such as x and A denote vectors and matrices respectively, although lower-case letters might also be used for scalars or functions (the distinction will be apparent from the context). We use bracketed indices to denote the time of the true system, i.e., the system is at state $x(t)$ at time t, subscripts denote prediction time indices within an MPC loop, i.e., x_t denotes the t^{th} predicted state, and superscripts to denote iteration steps of an optimization algorithm, i.e., x_t^k is the value of the t^{th} predicted state after the k^{th} iteration. To denote subsystem variables, we use square bracket notation, i.e. $[x]_i$ denotes the components of vector x that correspond to subsystem i. Calligraphic letters such as $\mathcal{S}$ denote sets, and in particular lowercase letters such as $\mathfrak{c}$ denote a subset of $\mathbb{Z}^+$, i.e. $\mathfrak{c} = \{1, ..., n\} \subset \mathbb{Z}^+$. Boldface lower and upper case letters such as $\mathbf{x}$ and $\mathbf{K}$ denote finite horizon signals and lower block triangular operators, respectively:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_T \end{bmatrix}, \ \mathbf{K} = \begin{bmatrix} K^{0,0} & & & \\ K^{1,1} & K^{1,0} & & \\ \vdots & \ddots & \ddots & \\ K^{T,T} & \dots & K^{T,1} & K^{T,0} \end{bmatrix},$$

where each $K^{i,j}$ is a matrix of compatible dimension. In this notation, $\mathbf{K}$ is the matrix representation of the convolution operation induced by a time varying controller $K_t(x_{0:t})$, so that $u_t = \mathbf{K}^{t,:}\mathbf{x}$, where $\mathbf{K}^{t,:}$ represents the t-th block-row of $\mathbf{K}$. We use $\mathbf{K}(\mathfrak{r}, \mathfrak{c})$ to denote the submatrix of $\mathbf{K}$ composed by the rows according to the set $\mathfrak{r}$ and columns according to the set $\mathfrak{c}$.

II. PROBLEM FORMULATION

In this section, we formulate the distributed MPC problem, and define the assumptions that we make throughout the remainder of the paper.

A. Distributed and Localized MPC formulation

We consider the discrete-time linear time invariant (LTI) system of the form

$$x(t+1) = Ax(t) + Bu(t) + w(t), \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state, $u(t) \in \mathbb{R}^p$ is the control input, and $w(t) \in \mathbb{R}^n$ is an exogenous disturbance. The system is composed of N interconnected subsystems, as defined by an interconnection topology – correspondingly, the state, control, and disturbance inputs can be suitably partitioned as $[x]_i$, $[u]_i$, and $[w]_i$, inducing a compatible block structure $[A]_{ij}$, $[B]_{ij}$ in the dynamics matrices (A, B). We model the interconnection topology of the system as a time-invariant[1] unweighted directed graph $\mathcal{G}(E, V)$, where each subsystem i is identified with a vertex $v_i \in V$ and an edge $e_{ij} \in E$ exists whenever $[A]_{ij} \neq 0$ or $[B]_{ij} \neq 0$.

Example 1: Consider the linear time-invariant system structured as a chain topology as shown in Figure 1. Each subsystem i is subject to the dynamics

$$[x(t+1)]_i = \sum_{j \in \{i, i\pm 1\}} [A]_{ij}[x(t)]_j + [B]_{ii}[u(t)]_i + [w(t)]_i.$$

As B is a diagonal matrix, coupling between subsystems is defined by the A matrix – thus, the adjacency matrix of the corresponding graph $\mathcal{G}$ coincides with the support of A.

As is standard, a model predictive controller is implemented by solving a series of finite horizon optimal control

[1] Although we restrict both the dynamics and the interconnection topology to be time-invariant, we believe that extending our results to time-varying dynamics and interconnection topologies will be straightforward, we leave this as future work.

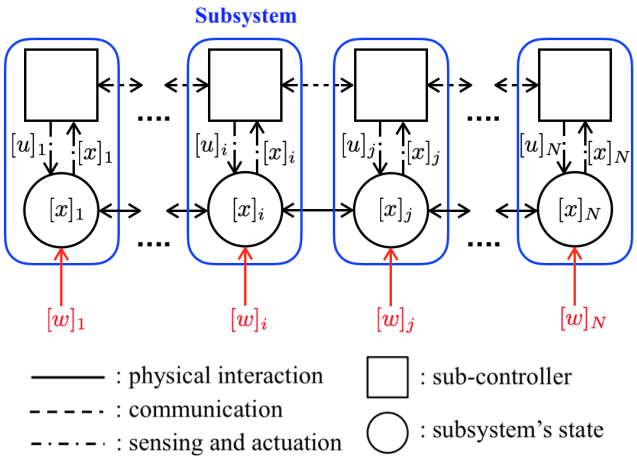

Fig. 1. Schematic representation of a system with a chain topology.

problems, with the problem solved at time t solved with initial condition $x_0 = x(t)$, over a prediction horizon T:

$$\begin{aligned} \min_{x_t, u_t} \quad & \sum_{t=0}^{T-1} f_t(x_t, u_t) + f_T(x_T) \\ & x_{t+1} = A_t x_t + B_t u_t, \quad t = 0, ..., T-1 \\ \text{s.t.} \quad & x_0 = x(t) \\ & x_T \in \mathcal{X}_T, \ x_t \in \mathcal{X}_t, \ u_t \in \mathcal{U}_t \quad t = 0, ..., T-1, \end{aligned} \tag{2}$$

where each of the problems takes the previous measurement of the state, x_0, as the initial condition.

Our goal is to define a distributed MPC controller that exploits the underlying structure of the dynamics (1) so that both synthesis, i.e., computing the solution to optimization problem (2), and implementation, i.e., applying the computed control policy, can be done in a distributed manner, while requiring only local exchange of information between sub-controllers. Specifically, we assume that the information exchange topology between sub-controllers matches that of the underlying system, and we further impose that only *local* information can be exchanged between sub-controllers, where a d-local information exchange constraint means that a sub-controller can only exchange information with its neighbors that are at most d-hops away. To formalize these ideas, we introduce the notion of d-outgoing and d-incoming sets for a given subsystem.

Definition 1: For a graph $\mathcal{G}(V, E)$, the *d-outgoing set* of subsystem i is $\mathbf{out}_i(d) := \{v_j \mid \mathbf{dist}(v_i \to v_j) \leq d \in \mathbb{N}\}$. Analogously, the *d-incoming set* of subsystem i is $\mathbf{in}_i(d) := \{v_j \mid \mathbf{dist}(v_j \to v_i) \leq d \in \mathbb{N}\}$. Note that $v_i \in \mathbf{out}_i(d) \cap \mathbf{in}_i(d)$ for all $d \geq 0$.

With this definition in mind, a d-local sub-controller information exchange topology with respect to the system topology graph $\mathcal{G}(E, v)$ can be represented by an unweighted graph $\mathcal{H}(E', V)$, where each sub-controller i is associated with the corresponding vertex of its subsystem $V_i \in V$, and an edge $e'_{ij} \in E'$ exists only if $j \in \mathbf{out}_i(d)$, i.e., sub-controller i can directly send information to sub-controller j only if it as at most d-hops away according to the system topology defined by $\mathcal{G}$. We show later that d can be viewed as a design parameter in that varying the size of the localized regions allows for a principled trade-off between the amount of coordination allowed between sub-controllers, the computational complexity of the resulting distributed MPC controller, and the performance that it achieves. Finally, we work with the assumption that no time delay is present in this local communication between subsystems. These information exchange constraints can be imposed on the solution to the MPC problem (2) by enforcing the additional constraint that

$$\begin{aligned} [u_t]_i = \gamma_{i,t} \big(& [x_{0:t}]_{j \in \mathbf{in}_i(d)}, [u_{0:t}]_{j \in \mathbf{in}_i(d)}, \\ & [A]_{j,k \in \mathbf{in}_i(d)}, [B]_{j,k \in \mathbf{in}_i(d)} \big), \end{aligned} \tag{3}$$

for all $t = 0, \ldots, T$ and $i = 1, \ldots, N$, where $\gamma_{i,t}$ is a measurable function of its arguments. As we will demonstrate in the next section, reformulating the MPC optimization problem as one over *system responses* allows for such information sharing constraints to be naturally, tractably, and scalably imposed, under suitable locality assumptions, which we introduce now.

Assumption 1: The objective function f_t in formulation (2) is such that $f_t(x) = \sum f_{t_i}([x]_{j \in \mathbf{in}_i(d)}, [u]_{j \in \mathbf{in}_i(d)})$. The constrain sets in formulation (2) are such that $x \in \mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_n$, where $x \in \mathcal{X}$ if and only if $[x]_{j \in \mathbf{in}_i(d)} \in \mathcal{X}_i$ for all i, and idem for $\mathcal{U}$.

Assumption 1 imposes that whenever two subsystems are coupled through either the constraints or the objective function, they must be within the local region of one another (d-incoming and d-outgoing sets). This is a natural assumption for large structured networks where couplings between subsystems tend to occur at a local scale. Moreover, in order to guarantee recursive feasibility of problem (2) we make the following standard assumption:

Assumption 2: The time horizon T of the MPC subroutine in (2) is sufficiently long to ensure recursive feasibility (Corollary 13.2, §13 in [13]).

In the next section, we show that under these assumptions, we can synthesize and implement a distributed and localized MPC controller by reformulating optimization problem (2) subject to the information exchange constraints (3) as one over structured *system responses* – to do so, we leverage the System Level Synthesis (SLS) framework.

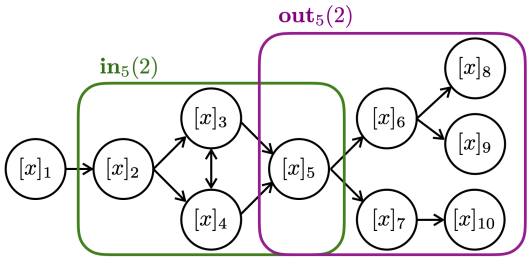

Fig. 2. Example of 2-ingoing and 2-outgoing sets for subsystem 5.

III. SYSTEM LEVEL SYNTHESIS BASED DLMPC

In this section, we introduce the relevant tools from the SLS framework, and show how it naturally allows for locality constraints [12] to be imposed on the system responses and corresponding controller implementation – for a more extensive overview of the SLS framework and its applications to distributed and robust control, please refer to [12].

A. Time Domain System Level Synthesis

Consider the dynamics of the system (1) evolving over a finite horizon $t = 0, ...T$, and let u_t be a causal time-varying state-feedback controller, i.e. $u_t = K_t(x_0, x_1, ..., x_t)$ where K_t is some linear map to be designed. [2]

Let Z be the block-downshift operator, i.e. a matrix with identity matrices along its first block sub-diagonal and zeros elsewhere, and define $A := \text{blkdiag}(A_0, A_1, ..., A_{T-1}, 0)$ and $B := \text{blkdiag}(B_0, B_1, ..., B_{T-1}, 0)$. This allows us to write the behavior of the system (1) over the horizon $t = 0, ..., T$ as

$$\mathbf{x} = Z(A + B\mathbf{K})\mathbf{x} + \mathbf{w}, \tag{4}$$

where $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{w}$ are the finite horizon signals corresponding to state, control input and disturbance respectively as stated in the Notation section. In particular, the initial condition x_0 is seen as the first component of the finite horizon signal of the disturbance, i.e. $\mathbf{w} = [x_0 \ w_0 \ ...w_T]^\mathsf{T}$.

The closed loop behavior of the system in (1) under the feedback law $\mathbf{K}$ can be entirely characterized by

$$\begin{aligned} \mathbf{x} &= (I - Z(A + B\mathbf{K}))^{-1}\mathbf{w} =: \mathbf{\Phi}_x\mathbf{w} \\ \mathbf{u} &= \mathbf{K}(I - Z(A + B\mathbf{K}))^{-1}\mathbf{w} =: \mathbf{\Phi}_u\mathbf{w}. \end{aligned} \tag{5}$$

The approach taken by SLS is to directly parameterize and optimize over the set of achievable closed loop maps (5) from the exogenous disturbance $\mathbf{w}$ to the state $\mathbf{x}$ and the control input $\mathbf{u}$, respectively.

Theorem 1 (Theorem 2.1, [12]): For the dynamics (1) evolving under the state-feedback policy $\mathbf{u} = \mathbf{K}\mathbf{x}$, for $\mathbf{K}$ a block-lower-triangular matrix, the following are true

1) The affine subspace
$$[I - ZA \quad -ZB] \begin{bmatrix} \mathbf{\Phi}_x \\ \mathbf{\Phi}_u \end{bmatrix} = I \tag{6}$$
parameterizes all possible system responses (5).
2) For any block-lower-triangular matrices $\{\mathbf{\Phi}_x, \mathbf{\Phi}_u\}$ satisfying (6), the controller $\mathbf{K} = \mathbf{\Phi}_u\mathbf{\Phi}_x^{-1}$ achieves the desired response (5) from $\mathbf{w} \mapsto (\mathbf{x}, \mathbf{u})$.

Theorem 1 allows us to reformulate an optimal control problem over state and input pairs $(\mathbf{x}, \mathbf{u})$ as an equivalent one over system responses $\{\mathbf{\Phi}_x, \mathbf{\Phi}_u\}$ – a detailed description of this transformation for several standard robust and optimal control problems is provided in [12]. For the MPC sub-problem (2), as no driving noise is present, we only have to account for the system response to the initial condition x_0, i.e. $\mathbf{w} = [x_0^\mathsf{T}, 0, ..., 0]^\mathsf{T}$. Hence, by equation (5), $\mathbf{x} = \mathbf{\Phi}[0]_x x_0$ and $\mathbf{u} = \mathbf{\Phi}[0]_u x_0$, where $\mathbf{\Phi}_x[0]$ and $\mathbf{\Phi}_u[0]$ denote the first block column of the block lower triangular response matrices $\mathbf{\Phi}_x[0]$ and $\mathbf{\Phi}_u[0]$ – in the sequel, we abuse notation and write $\mathbf{\Phi}_x[0] = \mathbf{\Phi}_x$, $\mathbf{\Phi}_u[0] = \mathbf{\Phi}_u$, as in the absence of driving noise, only the first block columns of the system responses need to be computed. This identification allows us to rewrite the MPC sub-problem (2) as

$$\begin{aligned} \min_{\mathbf{\Phi}_x, \mathbf{\Phi}_u} \quad & f(\mathbf{\Phi}_x x_0, \mathbf{\Phi}_u x_0) \\ \text{s.t.} \quad & Z_{AB} \begin{bmatrix} \mathbf{\Phi}_x \\ \mathbf{\Phi}_u \end{bmatrix} = I \\ & \mathbf{\Phi}_x x_0 \in \mathcal{X}^T, \ \mathbf{\Phi}_u x_0 \in \mathcal{U}^T, \\ & x_0 = x(t), \end{aligned} \tag{7}$$

where we let $Z_{AB} := [I - ZA \quad -ZB]$, and $\mathcal{X}^T := \otimes_{t=0}^{T-1}\mathcal{X} \otimes \mathcal{X}_T$, and similarly for $\mathcal{U}^T$, and f is suitably defined such that it is consistent with the objective function of problem (2). Clearly, if the original MPC sub-problem (2) is convex, then so is the SLS reformulation (7). Moreover, the control law $\mathbf{u}$ achieving the responses computed with optimization problem (7) can be implemented as

$$\mathbf{u} = \mathbf{\Phi}_u\hat{\mathbf{w}}, \quad \hat{\mathbf{x}} = (I - \mathbf{\Phi}_x)\hat{\mathbf{w}}, \quad \hat{\mathbf{w}} = \mathbf{x} - \hat{\mathbf{x}}, \tag{8}$$

where $\hat{\mathbf{x}}$ is the nominal state trajectory, and $\hat{\mathbf{w}} = Z\mathbf{w}$ is a delayed reconstruction of the disturbance. The advantage of this controller implementation, as opposed to $\mathbf{u} = \mathbf{\Phi}_u\mathbf{\Phi}_x^{-1}\mathbf{x}$, is that any structure imposed on $\{\mathbf{\Phi}_u, \mathbf{\Phi}_x\}$ translates directly to structure on the controller implementation (8).

Notice that in contrast to disturbance based MPC [14], which only optimizes over the map from disturbance to control input, the SLS based approach parameterizes the map from disturbance $\mathbf{w}$ to both state and control input. By explicitly enforcing localized structure on both maps $\{\mathbf{\Phi}_u, \mathbf{\Phi}_x\}$, local disturbances $[w]_i$ can be computed using only local information (by exploiting the structure $\mathbf{\Phi}_x$), and corresponding control actions can be implemented using only local information (by exploiting the structure $\mathbf{\Phi}_u$). We make this observation precise in the next subsection.

B. Locality in System Level Synthesis

Here we introduce the notion of d-localized system responses, and show that locality in system responses translates into locality of controller implementation. In the next section, we show that localized system responses further allow for localized computation of optimal control policies, i.e., for localized synthesis.

Definition 2: Let $[\mathbf{\Phi}_x]_{ij}$ be the submatrix of system response $\mathbf{\Phi}_x$ describing the map from disturbance $[w]_j$, which directly affects subsystem j, to $[x]_i$ the state of subsystem i. The map $\mathbf{\Phi}_x$ is *d-localized* if and only if for every subsystem j, $[\mathbf{\Phi}_x]_{ij} = 0 \ \forall \ i \notin \textbf{out}_j(d)$. The definition for *d-localized* $\mathbf{\Phi}_u$ is analogous but with perturbations to control action $[u]_i$ at subsystem i.

It then follows immediately from equation (8) that if the system responses are d-localized, then so is the controller

[2] We restrict ourselves to a linear map without loss of generality, as an affine control policy $u_t = K_t(x_{0:t}) + v_t$ can always be written as a linear policy acting on the augmented state $\tilde{x} = [x; 1]$.

implementation. In particular, by enforcing d-localized structure on $\mathbf{\Phi}_x$, only a corresponding local subset $[\hat{\mathbf{w}}]_{j\in\mathbf{in}_i(d)}$ of $\hat{\mathbf{w}}$ are necessary for subsystem i to compute its local disturbance estimate $[\hat{\mathbf{w}}]_i$, which ultimately means that only local communication is required to reconstruct the relevant disturbances for each subsystem. Similarly, if d-localized structure is imposed on $\mathbf{\Phi}_u$, then only a local subset $[\hat{\mathbf{w}}]_{j\in\mathbf{in}_i(d)}$ of the estimated disturbances $\hat{\mathbf{w}}$ are needed for each subsystem to compute its control action $[\mathbf{u}]_i$. Hence, each subsystem only needs to collect information from its d-incoming set to implement the control law defined by (8). Furthermore, such locality constraints are transparently enforced as additional subspace constraints in the SLS formulation (7).

Definition 3: A subspace $\mathcal{L}_d$ enforces a *d-locality constraint* if $\mathbf{\Phi}_x, \mathbf{\Phi}_u \in \mathcal{L}_d$ implies that $\mathbf{\Phi}_x$ is d-localized and $\mathbf{\Phi}_u$ is $(d+1)$-localized. A system (A, B) is then d-localizable if the intersection of $\mathcal{L}_d$ with the affine space of achievable system responses, defined by Z_{AB}, is non-empty.

We can now formulate the DLMPC sub-problem by suitably incorporating locality constraints into the SLS based MPC sub-problem (6).

$$\begin{aligned} \min_{\mathbf{\Phi}_x,\mathbf{\Phi}_u} \quad & f(\mathbf{\Phi}_x x_0, \mathbf{\Phi}_u x_0) \\ & Z_{AB}\begin{bmatrix} \mathbf{\Phi}_x \\ \mathbf{\Phi}_u \end{bmatrix} = I \\ \text{s.t.} \quad & \mathbf{\Phi}_x x_0 \in \mathcal{X}^T, \mathbf{\Phi}_u x_0 \in \mathcal{U}^T \\ & \mathbf{\Phi}_x, \mathbf{\Phi}_u \in \mathcal{L}_d, \\ & x_0 = x(t), \end{aligned} \tag{9}$$

where f, $\mathcal{X}^T$ and $\mathcal{U}^T$ are as before.

While it was not obvious how to impose locality constraints on information exchange in the original formulation of the MPC sub-problem (2), it is straightforward to do so via the locality constraints $\mathbf{\Phi}_x, \mathbf{\Phi}_u \in \mathcal{L}_d$ in the SLS formulation (9). We emphasize that imposing localized structure in both $\mathbf{\Phi}_x$ and $\mathbf{\Phi}_u$ is needed to ensure locality of the resulting controller implementation.

Remark 1: Notice that we are imposing $\mathbf{\Phi}_u$ to be $(d+1)$-localized because in order to localize the effects of a disturbance within the region of size d, the "boundary" controllers at distance $d+1$ should also know about the disturbance in order to take action. For more details the reader is referred to [12].

Remark 2: Note that although d-locality constraints can always be imposed as convex subspace constraints, not all systems are d-localizable. As we describe in the sequel, the locality diameter d can be viewed as design parameter, and for the remainder of the paper, we assume that there exists a $d << N$ such that the system (A, B) to be controlled is d-localizable. Although beyond the scope of this paper, all of the presented results extend naturally to systems that are approximately localizable using a robust variant of the SLS parameterization described in [15], [12].

Example 2: Consider the chain in Example 1, and suppose that we enforce a 1-locality constraint on the system responses: then $\mathbf{\Phi}_x$ is 1-localized and $\mathbf{\Phi}_u$ is 2-localized. Due to the chain topology, this is equivalent to enforcing a tridiagonal structure on $\mathbf{\Phi}_x$ and a pentadiagonal structure on $\mathbf{\Phi}_u$. The resulting 1-outgoing and 2-incoming sets at node i are then given by $\mathbf{out}_i(1) = \{i-1, i, i+1\}$ and $\mathbf{in}_i(2) = \{i-2, i-1, i, i+1, i+2\}$, as illustrated in Figure 3.

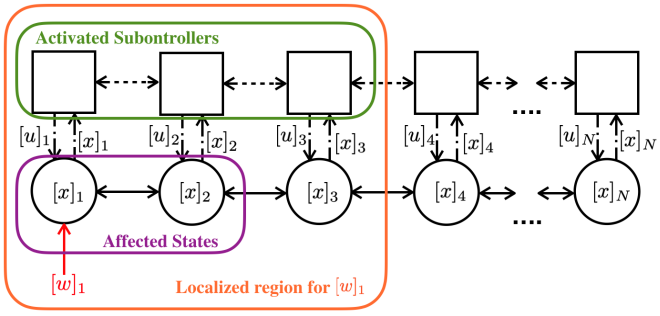

Fig. 3. Graphical representation of the scenario described in Example 2.

Notice that the size of the localized region d is a design variable and can be chosen from a range of values. In particular, when d equals the size of the network, formulation (7) synthesizes a centralized controller – thus the optimization problem is always feasible for such a d, and the control cost cannot be improved upon by any other controller. However, such a centralized implementation requires global communication and solving a large-scale optimal control problem, which may not be feasible. In order to reduce the communication and computational burden of synthesizing and implementing a control law, it is desirable that d be much smaller than the size of the network – however, restricting the system responses to be d-localized for a very small d can lead to a degradation in performance (much as enforcing too short a temporal horizon in deadbeat control can lead to a degradation in performance). Therefore, by appropriately selecting the locality parameter d, near optimal performance, as measured relative to a controller with global centralized communication & computation, can be achieved under a significantly reduced communication & computational burden.

It remains to demonstrate that imposing locality constraints also allows us to decompose optimization problem (9) into sub-problems, such that the sub-problem solved by sub-controller i requires only d-local information exchange and d-local system models. In the next section, we show how the localized MPC problem an be solved in such a d-localized manner by an ADMM based algorithm. The proposed method is applicable to any convex constraints and objective functions, so long as they only introduce direct coupling between d-local neighboring subsystems (Assumption 1). To the best of our knowledge, we are the first to propose a distributed MPC problem that can be exactly solved, and is applicable to general cooperative control tasks with dynamically coupled subsystems subject to coupling constraints [16].

IV. AN ADMM BASED DISTRIBUTED AND LOCALIZED SOLUTION

We start with a brief overview of the ADMM algorithm, and then show how it can be used to decompose the DLMPC sub-problem (9) into sub-problems that can be solved using only d-local information. For the sake of clarity, we introduce a simpler version of the algorithm first where only dynamical coupling is considered, which we then extend to the constraints and objective functions that introduce d-localized couplings, as defined in Assumption 1.

A. The Alternating Direction Method of Multipliers

The ADMM [17] algorithm has proved successful solving large-scale optimization problems that respect a certain partial-separability structure. In particular, given the following optimization problem

$$\begin{aligned} \min_{x,y} \quad & f(x) + g(y) \\ \text{s.t.} \quad & Ax + By = c, \end{aligned} \tag{10}$$

ADMM - in its scaled form - solves (10) with the following update rules:

$$\begin{aligned} x^{k+1} &= \arg\min_x f(x) + \frac{\rho}{2}\left\|Ax + By^k - c + z^k\right\|_2^2 \\ y^{k+1} &= \arg\min_y g(y) + \frac{\rho}{2}\left\|Ax^{k+1} + By - c + z^k\right\|_2^2 \\ z^{k+1} &= z^k + Ax^{k+1} + By^{k+1} - c. \end{aligned} \tag{11}$$

ADMM is particularly powerful when the iterate sub-problems defined in equation (11) can be solved in closed form, which is the case in many practically relevant cases [17], as this allows for rapid execution and convergence of the algorithm. Further, under mild assumptions ADMM enjoys strong and general convergence guarantees.

Theorem 2: Assume that extended real valued functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ are closed, proper, and convex. Moreover, assume that the unaugmented Lagrangian has a saddle point. Then, the ADMM iterates in equation (11) satisfy the following:

- **Residual convergence:** $r_t \to 0$ as $t \to \infty$, i.e. the iterates approach feasibility.
- **Objective convergence:** $f(x_t) + g(z_t) \to p^*$ as $t \to 0$, i.e. the objective function of the iterates approaches the optimal value.
- **Dual variable convergence:** $y_t \to y^*$ as $t \to \infty$, where y^* is a dual optimal point.

Hence, we impose the following additional assumptions so as to ensure that Theorem 2 is applicable to the considered problem.

Assumption 3: Problem (9) has a feasible solution in the relative interior of $\mathcal{X}^T$ and $\mathcal{U}^T$.

Assumption 4: The constraint sets $\mathcal{X}^T$ and $\mathcal{U}^T$ in formulation (9) are closed and convex. The objective function $f(\mathbf{\Phi} x_0)$ is a closed, proper, and convex function for all choices of $x_0 \neq 0$.

B. DLMPC without Coupling Constraints

We present here a simplified version of the algorithm that contains its main features. In the next subsection the general version of the algorithm will be introduced.

1) Derivation of the algorithm: In this subsection we illustrate how to decompose the MPC subroutine (9) by exploiting its separability. We begin by restricting ourselves to the case where neither the objective function nor the constraints introduce any coupling between subsystems. We present how the DLMPC problem (9) can be decomposed into local sub-problems that can be solved using only d-local information and system models. In particular, consider the DLPMPC sub-problem solved at time t:

$$\begin{aligned} \min_{\mathbf{\Phi}} \quad & f(\mathbf{\Phi} x_0) \\ \text{s.t.} \quad & Z_{AB}\mathbf{\Phi} = I,\ \mathbf{\Phi} x_0 \in \mathcal{P},\ \mathbf{\Phi} \in \mathcal{L}_d,\ x_0 = x(t) \end{aligned} \tag{12}$$

where, to lighten notation, we let $\mathbf{\Phi} := [\mathbf{\Phi}_x^{\mathsf{T}}\ \mathbf{\Phi}_u^{\mathsf{T}}]^{\mathsf{T}}$ and $\mathbf{\Phi} x_0 \in \mathcal{P} \iff \mathbf{\Phi}_x x_0 \in \mathcal{X}^T$ and $\mathbf{\Phi}_u x_0 \in \mathcal{U}^T$. Moreover, according to the statement above, neither the objective function nor the constraints can introduce any coupling between subsystems, i.e., that f is can be written as $f(\mathbf{x}, \mathbf{u}) = \sum f_i([\mathbf{x}]_i, [\mathbf{u}]_i)$, or equivalently, that $f(\mathbf{\Phi} x_0) = \sum f_i(\mathbf{\Phi}(\mathfrak{r}_i, :)x_0)$ since $[\mathbf{x}]_i = \mathbf{\Phi}(\mathfrak{r}_i, :)x_0$ where $\mathfrak{r}_i$ is the set of rows in $\mathbf{\Phi}$ corresponding to subsystem i, and equivalently for $[\mathbf{u}]_i$. Similarly, under the no coupling assumption, the constraints must satisfy that $\mathbf{x} \in \mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_n$ if and only if each $[\mathbf{x}]_i \in \mathcal{X}_i$, and idem for $\mathcal{U}$, hence $\mathbf{\Phi}(\mathfrak{r}_i, :)x_0 \in \mathcal{P}_i$.

These separability properties can be formalized through the following definitions, adapted from [11]:

Definition 4:

1) The functional $g(\mathbf{\Phi})$ is *column-wise separable* with respect to the partition $\mathfrak{c} = \{\mathfrak{c}_1, ..., \mathfrak{c}_p\}$ if it can be written as $g(\mathbf{\Phi}) = \sum_{j=1}^p g_j(\mathbf{\Phi}(:, \mathfrak{c}_j))$ for some functionals g_j for $j = 1, ..., p$. Equivalently, $g(\mathbf{\Phi})$ is *row-wise separable* with respect to the partition $\mathfrak{r}$ if it can be written as $g(\mathbf{\Phi}) = \sum_{j=1}^p g_j(\mathbf{\Phi}(\mathfrak{r}_j, :)$.
2) A constraint-set $\mathcal{P}$ is *column-wise separable* with respect to the partition $\mathfrak{c} = \{\mathfrak{c}_1, ..., \mathfrak{c}_p\}$ when $\mathbf{\Phi} \in \mathcal{P} \iff \mathbf{\Phi}(:, \mathfrak{c}_j) \in \mathcal{P}_j$ for $j = 1, ..., p$ is satisfied for some sets $\mathcal{P}_j$ for $j = 1, ..., p$. Equivalently, $\mathcal{P}$ is *row-wise separable* with respect to the partition $\mathfrak{r}$ if $\mathbf{\Phi} \in \mathcal{P} \iff \mathbf{\Phi}(\mathfrak{r}_j, :) \in \mathcal{P}_j$ for $j = 1, ..., p$.

It is easily seen that if all objective functions and constraints in an optimization problem are either column-wise (row-wise) separable with respect to a partition $\mathfrak{c}$ ($\mathfrak{r}$) of cardinality p, then the optimization problem can be trivially decomposed into p independent sub-problems. However, while our assumption of no dynamic coupling imposes that $f_{x_0}(\mathbf{\Phi}) = f(\mathbf{\Phi} x_0)$ and $\mathcal{P}$ are row-wise separable in the optimization variable $\mathbf{\Phi}$, the achievability constraint $Z_{AB}\mathbf{\Phi} = I$ is column-wise separable in the optimization variable $\mathbf{\Phi}$. As shown in [11], this partially-separable structure can be exploited within an ADMM based algorithm to reduce each ADMM iterate sub-problem (11) to a row or column-wise separable optimization problem, allowing the algorithm to

trivially decompose and be solved at scale.

Definition 5: An optimization problem is partially separable if it can be written as

$$\begin{aligned} \min_{\mathbf{\Phi}} \quad & g^{(r)}(\mathbf{\Phi}) + g^{(c)}(\mathbf{\Phi}) \\ \text{s.t.} \quad & \mathbf{\Phi} \in \mathcal{S}^{(r)} \cap \mathcal{S}^{(c)}, \end{aligned} \tag{13}$$

for row-wise separable $g^{(r)}$ and $\mathcal{S}^{(r)}$, and column-wise separable $g^{(c)}$ and $\mathcal{S}^{(c)}$.

Problem (12) is partially separable since $f(\mathbf{\Phi}x_0)$ and $\mathcal{P}$ are row-wise separable and $Z_{AB}\mathbf{\Phi} = I$ is column-wise separable with respect to $\mathbf{\Phi}$. Therefore, we reformulate the DLMPC sub-problem (9) so that is of the form (13):

$$\begin{aligned} \min_{\mathbf{\Phi},\mathbf{\Psi}} \quad & f(\mathbf{\Phi}x_0) \\ \text{s.t.} \quad & Z_{AB}\mathbf{\Psi} = I,\ \mathbf{\Phi}x_0 \in \mathcal{P},\ \{\mathbf{\Phi},\mathbf{\Psi}\} \in \mathcal{L}_d,\ \mathbf{\Phi} = \mathbf{\Psi}. \end{aligned} \tag{14}$$

Notice that by duplicating the decision variable, we can decompose the original DLMPC sub-problem (9) into a column-wise separable iterate sub-problem in $\mathbf{\Phi}$, and a row-wise seperable iterate sub-problem in $\mathbf{\Psi}$ – thus optimization problem (14) is partially separable, and is amenable to a distributed solution. In particular, the ADMM algorithm applied to problem (14) gives

$$\mathbf{\Phi}^{k+1} = \left\{ \begin{array}{ll} \underset{\mathbf{\Phi}}{\text{argmin}} & f(\mathbf{\Phi}x_0) + \dfrac{\rho}{2}\left\|\mathbf{\Phi} - \mathbf{\Psi}^k + \mathbf{\Lambda}^k\right\|_F^2 \\ \text{s.t.} & \mathbf{\Phi}x_0 \in \mathcal{P},\ \mathbf{\Phi} \in \mathcal{L}_d \end{array} \right\} \tag{15a}$$

$$\mathbf{\Psi}^{k+1} = \left\{ \begin{array}{ll} \underset{\mathbf{\Psi}}{\text{argmin}} & \left\|\mathbf{\Phi}^{k+1} - \mathbf{\Psi} + \mathbf{\Lambda}^k\right\|_F^2 \\ \text{s.t.} & Z_{AB}\mathbf{\Psi} = I,\ \mathbf{\Psi} \in \mathcal{L}_d \end{array} \right\} \tag{15b}$$

$$\mathbf{\Lambda}^{k+1} = \mathbf{\Lambda}^k + \mathbf{\Phi}^{k+1} - \mathbf{\Psi}^{k+1}. \tag{15c}$$

The squared Frobenius norm is both row-wise and column-wise separable. Therefore, the resulting iterate sub-problems in (15) are separable: iterate sub-problem (15a) is row-wise separable with respect to the row partition $\mathfrak{r}$ induced by the sub-system-wise partitions of the state and control inputs, $[x]_i$ and $[u]_i$, iterate sub-problem (15b) is column-wise separable with respect to the column partition induced in a analogous manner, and iterate sub-problem (15c) is component-wise separable. Hence, each of the iterate sub-problems described in (15) can be decomposed into column, row, or element-wise sub-problems that can solved independently and in parallel, with each sub-controller i computing the solution to its component of the row or column-wise partition. Moreover, by enforcing that the system responses be d-localized, i.e., that $\mathbf{\Phi}_x, \mathbf{\Phi}_u \in \mathcal{L}_d$, the resulting sub-problem variables are sparse, allowing for a significant reduction in the dimension of the local sub-problem. For example, when considering the column-wise sub-problem evaluated at subsystem j, the ith row of the jth sub-system column partitions of $\mathbf{\Phi}_x(:, \mathfrak{c}_j)$ and $\mathbf{\Phi}_u(:, \mathfrak{c}_j)$) is nonzero only if $i \in \cup_{k\in\mathbf{out}_j(d)}\mathfrak{r}_k$ and $i \in \cup_{k\in\mathbf{out}_j(d+1)}\mathfrak{r}_k$, respectively. In particular, sub-controller i must solve the following sub-problems:

$$[\mathbf{\Phi}]_{i_r}^{k+1} = \left\{ \begin{array}{ll} \underset{[\mathbf{\Phi}]_{\mathbf{i_r}}}{\text{argmin}} & f_i([\mathbf{\Phi}]_{i_r}[x_0]_{i_r}) + \dfrac{\rho}{2}\left\|[\mathbf{\Phi}]_{i_r} - [\mathbf{\Psi}]_{i_r}^k + [\mathbf{\Lambda}]_{i_r}^k\right\|_F^2 \\ \text{s.t.} & [\mathbf{\Phi}]_{i_r}[x_0]_{i_r} \in \mathcal{P}_i \end{array} \right\} \tag{16a}$$

$$[\mathbf{\Psi}]_{i_c}^{k+1} = \left\{ \begin{array}{ll} \underset{[\mathbf{\Psi}]_{i_c}}{\text{argmin}} & \left\|[\mathbf{\Phi}]_{i_c}^{k+1} - [\mathbf{\Psi}]_{i_c} + [\mathbf{\Psi}]_{i_c}^k\right\|_F^2 \\ \text{s.t.} & [Z_{AB}]_{i_c}[\mathbf{\Psi}]_{i_c} = [I]_{i_c} \end{array} \right\} \tag{16b}$$

$$[\mathbf{\Lambda}]_{i_r}^{k+1} = [\mathbf{\Lambda}]_{i_r}^k + [\mathbf{\Phi}]_{i_r}^{k+1} - [\mathbf{\Psi}]_{i_r}^{k+1}, \tag{16c}$$

where to lighten notational burden, we let $[\mathbf{\Phi}]_{i_r} := \mathbf{\Phi}(\mathfrak{s}_{\mathfrak{r}_i}, \mathfrak{r}_i)$, where the set $\mathfrak{r}_i$ represents the set of rows that the controller i is solving for, and the corresponding set $\mathfrak{s}_{\mathfrak{r}_i}$ is the set of columns associated to the rows in $\mathfrak{r}_i$ by the locality constraints $\mathcal{L}_d$. An equivalent argument applies to $[\mathbf{\Phi}]_{i_c} := \mathbf{\Phi}(\mathfrak{c}_i, \mathfrak{s}_{\mathfrak{c}_i})$ where the set $\mathfrak{c}_i$ represents the set of columns that the controller i is solving for, and the corresponding set $\mathfrak{s}_{\mathfrak{c}_i}$ is the set of columns associated to the rows in $\mathfrak{c}_i$. For example, when considering the row-wise sub-problem (16a) evaluated at subsystem i, the jth column of the ith sub-system row partition $\mathbf{\Phi}_x(\mathfrak{r}_i, :)$ and $\mathbf{\Phi}_u(\mathfrak{r}_i, :)$ is nonzero only if $j \in \mathbf{in}_j(d)$ and $j \in \mathbf{in}_j(d+1)$, respectively.

Informally, sets $s_{\mathfrak{r}_i}$ and $s_{\mathfrak{c}_i}$ are the collection of optimization variables contained within the localized region specified by the component of the locality constraint $\mathcal{L}_d$ that restricts the behavior of subsystem i. It follows that subsystem i only requires a corresponding subset of the local sub-matrices $[A]_{k,\ell}, [B]_{k,\ell}$ to solve its respective sub-problem. Thus by exploiting the sparsity of the underlying dynamics (A, B) and the enforced d-locality constraints on the system responses, we greatly reduce the complexity of the column-wise and row-wise sub-problems solved by each sub-controller. All column and row subsets described above can be found in a systematic way via the Dimension Reduction Algorithm described in Appendix A of [11].

Remark 3: Minimization (16b) can be solved in closed form:

$$\begin{aligned} [\mathbf{\Psi}]_{i_c}^{k+1} = \left([\mathbf{\Phi}]_{i_c}^{k+1} + [\mathbf{\Lambda}]_{i_c}^k\right) + [Z_{AB}]_{i_c}^+\Big([I]_{i_c} - \\ - [Z_{AB}]_{i_c}\left([\mathbf{\Phi}]_{i_c}^{k+1} + [\mathbf{\Lambda}]_{i_c}^k\right)\Big), \end{aligned} \tag{17}$$

where $[Z_{AB}]_{i_c}^+$ denotes the pseudo-inverse of $[Z_{AB}]_{i_c}$. We note that this pseudo-inverse can be computed once off-line, reducing the evaluation of update step (17) to matrix multiplication.

Notice that in general $\mathfrak{r}_i \subset \mathfrak{s}_{\mathfrak{c}_i}$ and $\mathfrak{c}_i \subset \mathfrak{s}_{\mathfrak{r}_i}$. Hence, each subsystem i is computing updates for the sub-matrix $\mathbf{\Phi}(\mathfrak{s}_{\mathfrak{r}_i}, c_i)$ and the sub-matrix $\mathbf{\Phi}(\mathfrak{r}_i, s_{\mathfrak{c}_i})$ of the global system response variables $\mathbf{\Phi}$ and $\mathbf{\Psi}$. In particular, for sub-system i to solve its local iterate sub-problems (16), information sharing among subsystems is needed. However, as we impose d-locality constraints on the system responses, information

only needs to be collected from d-hop neighbors. Similarly, only a d-local subset of the MPC sub-problem initial condition $x_0 = x(t)$ is needed to solve the local iterate sub-problems (16).

Algorithm 1 summarizes the sub-system wise implementation of the ADMM based solution to the DLMPC sub-problem (9) for systems with no couplings in the constraints and objective functions. Note that in the final step of Algorithm 1, we let $[x_0]_{\mathfrak{s}_{\mathfrak{r}_i}}$ denote the subset of elements of x_0 associated with the columns in $\mathfrak{s}_{\mathfrak{r}_i}$, such that $[\Phi_u^{0,0}]_{i_r} x_0 = [\Phi_u^{0,0}]_{i_r} [x_0]_{\mathfrak{s}_{\mathfrak{r}_i}}$.

Algorithm 1 Subsystem i DLMPC implementation, no coupling constraints

1: **input:** convergence tolerance parameters $\epsilon_p > 0$, $\epsilon_d > 0$
2: Measure local state $[x(t)]_i$.
3: Share the measurement with $\mathbf{out}_i(d)$.
4: Solve optimization problem (16b).
5: Share $[\mathbf{\Phi}]_{i_r}^{k+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{\Phi}]_{j_r}^{k+1}$ from $\mathbf{in}_i(d)$ and build $[\mathbf{\Phi}]_{i_c}^{k+1}$.
6: Solve optimization problem (16a) via the closed form solution (17).
7: Share $[\mathbf{\Psi}]_{i_c}^{k+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{\Phi}]_{j_c}^{k+1}$ from $\mathbf{in}_i(d)$ and build $[\mathbf{\Psi}]_{i_r}^{k+1}$.
8: Perform the multiplier update step (16c).
9: Check convergence as $\left\|[\mathbf{\Phi}]_{i_c}^{k+1} - [\mathbf{\Psi}]_{i_c}^{k+1}\right\|_F \leq \epsilon_p$ and $\left\|[\mathbf{\Psi}]_{i_c}^{k+1} - [\mathbf{\Psi}]_{i_c}^{k}\right\|_F \leq \epsilon_d$.
10: If converged, apply computed control action $[u_0]_i = [\Phi_u^{0,0}]_{i_r} [x_0]_{\mathfrak{s}_{\mathfrak{r}_i}}$, and return to step 2, otherwise return to step 4.

This algorithm is run in parallel by all the sub-controllers, and requires only information exchange with d-local neighbors, and only d-local sub-matrices of (A, B).

2) Computational complexity of the algorithm: The computational complexity of the algorithm is determined by update steps 4, 6 and 8. In particular, steps 6 and 8 can be directly solved in closed form, reducing their evaluation to the multiplication of matrices of dimension $O(d^2T)$. In certain cases, step 4 can also be computed in closed form if a proximal operator exists for the formulation. For instance this is true if it reduces to quadratic convex cost function subject to affine equality constraints. Regardless, each local iterate sub-problem is over $O(d^2T)$ optimization variables subject to $O(dT)$ constraints, leading to a significant computational saving when $d << N$. The communication complexity - as determined by steps 3, 5 and 7 - is limited to the local exchange of information between d-local neighbors.

3) Convergence of the algorithm: One can show convergence by leveraging Theorem 2.

Corollary 1: Algorithm 1 satisfies residual convergence, objective convergence and dual variable convergence as defined in Theorem 2.

Proof: Algorithm 1 is built upon algorithm (16), which is merely algorithm (15) after exploiting locality. Thus to prove Corollary 1 we only need to show that the ADMM algorithm (15) satisfies the assumptions in Theorem 2.

Define the extended-real-value functional $h(\mathbf{\Phi})$ by

$$h(\mathbf{\Phi}) = \begin{cases} \mathbf{\Phi} x_0 & \text{if } Z_{AB}\mathbf{\Phi} = I,\ \mathbf{\Phi} x_0 \in \mathcal{P},\ \mathbf{\Phi} \in \mathcal{L}_d \\ \infty & \text{otherwise.} \end{cases}$$

The constrained optimization in (14) can equivalently be written in terms of $h(\mathbf{\Phi})$ with the constraint $\mathbf{\Phi} = \mathbf{\Psi}$.

$$\min_{\mathbf{\Phi},\mathbf{\Psi}} \quad h(\mathbf{\Phi}) \text{ s.t. } \quad \mathbf{\Phi} = \mathbf{\Psi}.$$

Notice that by Assumption 4, $f(\mathbf{\Phi} x_0)$ is closed, proper, and convex, and $\mathcal{P}$ is a closed and convex set. Moreover, the remaining constraints $Z_{AB}\mathbf{\Phi} = I$ and $\mathbf{\Phi} \in \mathcal{L}_d$ are also closed and convex. Hence, $h(\mathbf{\Phi})$ is closed, proper, and convex. It only remains to show that the Lagrangian has a saddle point. This condition is equivalent to showing that strong duality holds [18]. By Assumption 3, Slater's condition is automatically satisfied, and therefore the Lagrangian of the problem has a saddle point. Since both conditions of Theorem 2 are satisfied, the ADMM algorithm in (15) satisfies residual convergence, objective convergence and dual variable convergence as defined in Theorem 2. Since Algorithm 1 results from leveraging (15), guaranteeing convergence of algorithm (15) automatically guarantees convergence of Algorithm 1. ■

C. DLMPC subject to localized coupling constraints

Building on Algorithm 1, we now show how DLMPC can be extended to allow for coupling between subsystems by the constraints and objective function, so long as the coupling is compatible with the d-localized constraints being imposed, i.e., so long as the objective function and constraints satisfy Assumption 1.

1) Derivation of the algorithm: Consider the DLMPC sub-problem (12), where the objective function and constraints satisfy the locality properties imposed by Assumption 1. Due to this local coupling, the problem is no longer partially separable – however, we may still rewrite it as

$$\begin{aligned} \min_{\mathbf{X},\mathbf{\Phi}} \quad & f(\mathbf{X}) \\ & \mathbf{X} = \mathbf{\Phi} x_0, \\ \text{s.t.} \quad & Z_{AB}\mathbf{\Phi} = I \\ & \mathbf{X} \in \mathcal{P},\ \mathbf{\Phi} \in \mathcal{L}_d. \end{aligned} \tag{18}$$

The first constraint is row-wise separable for $\mathbf{\Phi}$ while the second is column-wise separable in $\mathbf{\Phi}$. Applying the same variable duplication process as above and applying ADMM yields

$$[\mathbf{\Phi}^{k+1},\ \mathbf{X}^{k+1}] = \left\{ \begin{array}{ll} \underset{\mathbf{\Phi},\mathbf{X}}{\text{argmin}} & f(\mathbf{X}) + \frac{\rho}{2}\left\|\mathbf{\Phi} - \mathbf{\Psi}^k + \mathbf{\Lambda}^k\right\|_F^2 \\ \text{s.t.} & \mathbf{X} = \mathbf{\Phi} x_0,\ \mathbf{X} \in \mathcal{P},\ \mathbf{\Phi} \in \mathcal{L}_d \end{array} \right\} \tag{19a}$$

$$\mathbf{\Psi}^{k+1} = \left\{ \begin{array}{ll} \underset{\mathbf{\Psi}}{\text{argmin}} & \left\|\mathbf{\Phi}^{k+1} - \mathbf{\Psi} + \mathbf{\Lambda}^k\right\|_F^2 \\ \text{s.t.} & Z_{AB}\mathbf{\Psi} = I,\ \mathbf{\Psi} \in \mathcal{L}_d \end{array} \right\} \tag{19b}$$

$$\mathbf{\Lambda}^{k+1} = \mathbf{\Lambda}^k + \mathbf{\Phi}^{k+1} - \mathbf{\Psi}^{k+1} \tag{19c}$$

While the iterate sub-problems (19b) and (19c) enjoy column-wise and element-wise separability, iterate sub-problem (19a) is subject to local coupling due to the objective function f and constraint $\mathbf{X} \in \mathcal{P}$. In order to solve sub-problem iterate (19a) in a manner that respects the d-localized communication constraints, we propose an ADMM based consensus-like algorithm, similar to that used in [19]. Hence, the solution to iterate sub-problem (19a) is obtained by having each subsystem i solve

$$[[\mathbf{\Phi}]_{i_r}^{k+1,n+1},\ [\mathbf{X}]_{i_s}^{n+1}] = \left\{ \begin{array}{ll} & f_i(\mathbf{X}) + \frac{\rho}{2}\left\|[\mathbf{\Phi}]_{i_r} - [\mathbf{\Psi}]_{i_r}^n + [\mathbf{\Lambda}]_{i_r}^n\right\|_F^2 \\ \underset{[\mathbf{\Phi}]_{i_r},[\mathbf{X}]_{i_s}}{\text{argmin}} & + \frac{\mu}{2}\sum_{j\in\mathbf{in}_i(d)}\left\|[\mathbf{X}]_j^{n+1} - [\mathbf{Z}]_i + [\mathbf{Y}]_{ij}^n\right\|_F^2 \\ \text{s.t.} & [\mathbf{X}]_i = [\mathbf{\Phi}]_{i_r}[x_0]_{i_r},\ [\mathbf{X}]_{i_s} \in \mathcal{P}_i \end{array} \right\} \tag{20a}$$

$$[\mathbf{Z}]_i^{n+1} = \frac{1}{|\mathbf{in}_i(d)|}\sum_{j\in\mathbf{in}_i(d)}\left\|[\mathbf{X}]_j^{n+1} + [\mathbf{Y}]_{ij}^n\right\|_F^2 \tag{20b}$$

$$[\mathbf{Y}]_{ij}^{n+1} = [\mathbf{Y}]_{ij}^n + [\mathbf{X}]_i^{n+1} - [\mathbf{Z}]_j^{n+1}, \tag{20c}$$

where the notation used is as in (16). In particular $[\mathbf{X}]_{i_s}$ is the concatenation of components $[\mathbf{X}]_j$ satisfying $j \in \mathbf{in}_i(d)$, whereas $[\mathbf{X}]_i$ is restricted to only those components of $[\mathbf{X}]_{i_s}$ needed by subsystem i to compute $[\mathbf{\Phi}]_{i_r}x_0 = [\mathbf{\Phi}]_{i_r}[x_0]_{\mathfrak{s}_{\mathfrak{r}_i}}$. After reaching consensus in equations (20), one can use $[\mathbf{\Phi}]_{i_r}^{k+1}$ in algorithm (16). Therefore by solving iterate sub-problem (16a) using the ADMM based consensus-like updates (20), we are able to accommodate d-local coupling introduced in the constraints and objective function while still only exchanging information with d-local neighbors. The rest of the analysis follows just as in the previous subsection.

Algorithm 2 summarizes the general DLMPC algorithm as implemented at subsystem i.

2) Computational complexity and convergence guarantees: The presence of the coupling inevitably results in an increase of the computation and communication complexity of the algorithm. The computational complexity of the algorithm is now determined by steps 4, 6, 8, 11 and 13. All of these except for step 4 can be solved in closed form. Once again, by Assumption 1 all sub-problems are over $O(d^2T)$ optimization variables and $O(dT)$ constraints, so the complexity does not increase with the size of the network. There is however an increased computational burden due to the nested consensus-like algorithm used to solve iterate sub-problem (19a), which leads to an increase in the number of iterations needed for convergence. This also results in increased communication between subsystems, as local information exchange is needed as part of the consensus-like step as well. However, once again this exchange is limited to within a d-local subset of the system, resulting in small consensus problems that converge quickly, as we illustrate empirically in the next section.

Since Algorithm 2 is identical to Algorithm 1 save for the approach to solving the first iterate sub-problem, convergence follows from a similar argument as that used to prove Corollary 2.1.

Algorithm 2 Subsystem i implementation of DLMPC general subject to localized coupling

1: **input:** convergence tolerance parameters, ϵ_p, ϵ_d, $\epsilon_x > 0$.
2: Measure local state $[x_0]_i$.
3: Share measurement with $\mathbf{out}_i(d)$.
4: Solve optimization problem (20a).
5: Share $[\mathbf{X}]_i^{n+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{X}]_j^{n+1}$ from $\mathbf{in}_i(d)$.
6: Perform update (20b).
7: Share $[\mathbf{Z}]_i^{n+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{Z}]_j^{n+1}$ from $\mathbf{in}_i(d)$.
8: Perform update (20c).
9: If $\left\|[\mathbf{X}]_i^{n+1} - [\mathbf{Z}]_i^{n+1}\right\|_F < \epsilon_x$ go to step 10, otherwise return to step 4.
10: Share $[\mathbf{\Phi}]_{i_r}^{k+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{\Phi}]_{j_r}^{k+1}$ from $\mathbf{in}_i(d)$and build $[\mathbf{\Phi}]_{i_c}^{k+1}$.
11: Solve optimization problem (16a) via the closed form solution (17).
12: Share $[\mathbf{\Psi}]_{i_c}^{k+1}$ with $\mathbf{out}_i(d)$. Receive the corresponding $[\mathbf{\Phi}]_{j_c}^{k+1}$ from $\mathbf{in}_i(d)$ and build $[\mathbf{\Psi}]_{i_r}^{k+1}$.
13: Perform the multiplier update (16c).
14: Check convergence as $\left\|[\mathbf{\Phi}]_{i_c}^{k+1} - [\mathbf{\Psi}]_{i_c}^{k+1}\right\|_F \leq \epsilon_p$ and $\left\|[\mathbf{\Psi}]_{i_c}^{k+1} - [\mathbf{\Psi}]_{i_c}^{k}\right\|_F \leq \epsilon_d$.
15: If converged, apply computed control action $[u_0]_i = [\Phi_u^{0,0}]_{i_r}[x_0]_{\mathfrak{s}_{\mathfrak{r}_i}}$, and return to step 2, otherwise return to step 4.

V. SIMULATION EXPERIMENTS

In this section we illustrate the benefits of DLMPC as applied to large-scale distributed systems. To do so, we consider two case-studies. In the first, we aim to illustrate how the proposed algorithm accounts for general convex coupling objective functions and constraints. In the second, we empirically characterize the computational complexity properties of DLMPC. All code needed to replicate these experiments is available at `https://github.com/camoalon/2020ACC_DLMPC`.

A. Optimization features

In order to illustrate how the presented algorithm can accommodate local coupling between subsystems as introduced by the objective function and constraints, we present a dynamical system consisting of a chain of four pendulums coupled through a spring ($1N/m$) and a damper ($3Ns/m$). Each pendulum is modeled as a two-state subsystem described by its angle θ and angular velocity $\dot{\theta}$. Each of the pendulums can actuate its velocity. The simulations are done with a prediction horizon of $T = 10s$, and we impose a localized region of size $d = 1$ subsystems. The initial condition is arbitrarily generated with MATLAB $\mathrm{rng}(2020)$.

We compare the control performance achieved by solutions to the DLMPC sub-problem (12) computed as a centralized problem using the Gurobi solver [20] and CVX interpreter [21], [22], and using Algorithm 1 or 2 (solid line),

as appropriate, in Figure 4. In particular, we plot the evolution of the position of the first two pendulums under different control objectives and constraints. In scenario 1 we consider the quadratic cost $f(\mathbf{x},\mathbf{u}) = \sum_{i=1}^{4} \|[\mathbf{x}]_i\|_2^2 + \|[\mathbf{u}]_i\|_2^2$, and have no additional constraints. In Scenario 2 we consider a quadratic cost coupling the angle of adjacent pendulums, i.e. the control objective is a sum of functionals of the form $f([\theta]_i,[u]_i) = ([\theta]_i - \frac{1}{2}\sum[\theta]_j)^2 + [\dot{\theta}]_i^2 + [u]_i^2$. Finally, Scenario 3 uses the same objective function as Scenario 2 and further constrains the maximal allowable deviation between subsystem angles, i.e., $|[\theta]_i - [\theta]_j| \leq 0.05$, for all $t > 2$.

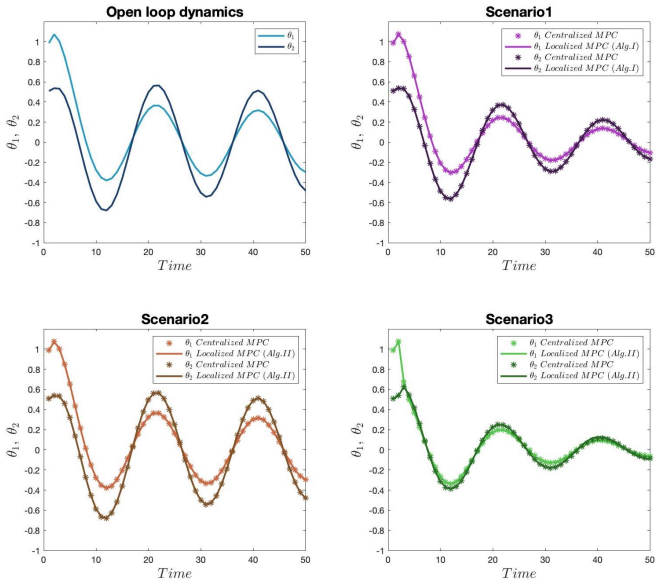

Fig. 4. The evolution of the position of the first two pendulums in open loop is shown in the top left figure, whereas the top right most figure shows the position of the first two pendulums under MPC control with a quadratic penalty on state and input, and no constraints (scenario 1). The bottom left figure shows the position of the first two pendulums when the performance objective couples the angle of adjacent pendulums. On the bottom right, the position of the first two pendulums when the performance objective and the constraints couple adjacent pendulums.

B. Per subsystem computational complexity

Here we show how Algorithms 1 and 2 allow DLMPC to be applied to large-scale systems, and further illustrate how the size d of the localized regions can be used to optimally trade-off between closed loop performance and computational complexity. Here, we let the substem dynamics be described by

$$[x(t+1)]_i = [A]_{ii}[x(t)]_i + \sum_{j \in \mathbf{in}_i(d)} [A]_{ij}[x(t)]_j + [B]_{ii}[u(t)]_i$$

where

$$[A]_{ii} = \begin{bmatrix} 1 & 0.1 \\ -0.3 & 0.7 \end{bmatrix}, \; [A]_{ij} = \begin{bmatrix} 0 & 0 \\ 0.1 & 0.1 \end{bmatrix}, \; [B]_{ii} = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}.$$

The MPC horizon is $T = 5$ and the locality parameter is set to $d = 3$ subsystems. We present four different scenarios that encompass different degrees of computational complexity of Algorithms 1 and 2:

- Case 1: per subsystem separable quadratic cost and no constraints.
- Case 2: per subsystem separable quadratic cost and per subsystem separable constraints.
- Case 3: quadratic cost coupling d-local subsystems and no constraints.
- Case 4: quadratic cost and polytopic constraints coupling d-local subsystems.

The computational complexity of each of the cases is determined by (i) if the row-wise iteration sub-problem can be solved in closed form, and (ii) if Algorithm 1 or 2 is needed – we summarize these properties for the four cases described above in Table I.

Case	Algorithm	Computation of step 3
1	1	Closed form
2	1	Needs minimization solver
3	2	Closed form
4	2	Needs minimization solver

TABLE I

As in [23], we characterize the runtime per state and MPC iteration of the DLMPC algorithms. In the leftmost plot of Figure 5, we fix the locality parameter as $d = 1$, and demonstrate that the runtime of both Algorithms 1 and 2 does not increase with the size of the network, assuming each sub-system is solving their sub-problems in parallel. These observations are consistent with those of [23] where the same trend was noted. The slight increase in runtime - in particular for Cases 3 and 4 - we conjecture is due to the introduced coupling, as the more subsystems are coupled together the longer it takes for the consensus-like sub-routine to converge. In any case, the increase in runtime does not seem to be significant and appears to level off for sufficiently large networks.

In the rightmost plot of Figure 5, we fix the number of systems to $N = 10$, and explore the effect of the size of the localized region d on computational complexity. While a larger localized region d can lead to improved performance, as a broader set of subsystems can coordinate their actions directly, it also leads to an increase in computational complexity, as the number of optimization variables per subproblem scales as $O(d^2T)$. Moreover, a larger localized region results in larger consensus-like problems being solved as a sub-routine in Algorithm 2, further contributing to a larger runtime. Thus by choosing the smallest localization parameter d such that acceptable performance is achieved, the designer can tradeoff between computational complexity and closed loop performance in a principled way. This further highlights the importance of exploiting the underlying structure of the dynamics, which allow us to enforce locality constraints on the system responses, and consequently, on the controller implementation.

VI. CONCLUSIONS

We defined and analyzed the Distributed and Localized MPC (DLMPC) problem by leveraging the SLS framework.

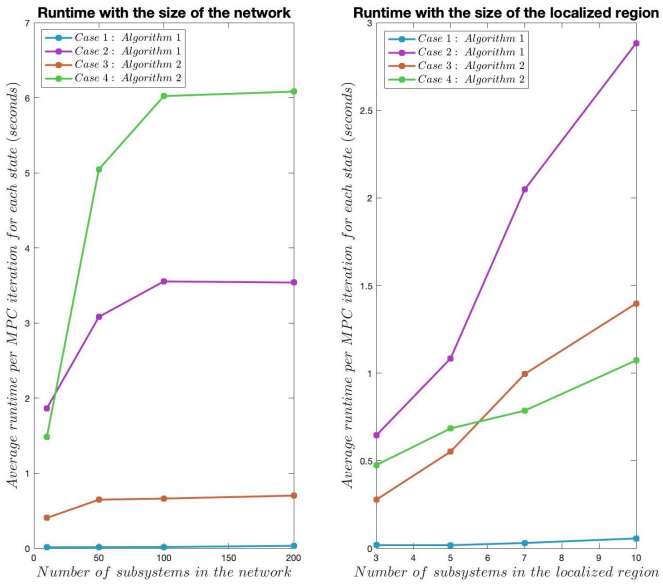

Fig. 5. On the left, the runtime of each of the four different cases for different network sizes. On the right, the runtime of each of the four different cases for different sizes of the localized region.

This framework has allowed us to naturally take into account the communication structure of the network by imposing locality constraints. We further showed that when locality is combined with mild assumptions on the separability structure of the objective functions and constraints the problem, an ADMM based solution to the DLMPC sub-problems can be implemented that requires only local information exchange and system models, making the approach suitable for large-scale distributed systems. Moreover, our approach is, to the best of our knowledge, the first that can accommodate constraints and objective functions that introduce local coupling between subsystems. In future work, we plan to develop robust variants of DLMPC that can accommodate both additive perturbations and model uncertainty. We will also extend our approach to approximately localizable systems by leveraging the robust SLS parameterization introduced in [15]. We will also explore whether locality constraints allow for a scalable computation of robust invariant sets for large-scale distributed systems, as well as their implications on the complexity of (approximate) explicit MPC approaches. Finally, it is of interest to extend the results presented in this paper to information exchange topologies defined in terms of both sparsity and delays – while the SLS framework naturally allows for delay to be imposed on the implementation structure of a distributed controller, it is less clear how to incorporate such constraints in a distributed optimization scheme.

REFERENCES

[1] M. Rotkowitz and S. Lall, "A Characterization of Convex Problems in Decentralized Control," *IEEE Transactions on Automatic Control*, vol. 51, no. 2, pp. 274–286, Feb. 2006.

[2] R. Scattolini, "Architectures for distributed and hierarchical Model Predictive Control – A review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, May 2009. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0959152409000353

[3] A. Venkat, I. Hiskens, J. Rawlings, and S. Wright, "Distributed MPC Strategies With Application to Power System Automatic Generation Control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1192–1206, Nov. 2008. [Online]. Available: http://ieeexplore.ieee.org/document/4497237/

[4] Y. Zheng, S. Li, and H. Qiu, "Networked Coordination-Based Distributed Model Predictive Control for Large-Scale System," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 991–998, May 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6199971/

[5] P. Giselsson and A. Rantzer, "On feasibility, stability and performance in distributed model predictive control," *arXiv:1302.1974 [math]*, Feb. 2013, arXiv: 1302.1974. [Online]. Available: http://arxiv.org/abs/1302.1974

[6] C. Conte, C. N. Jones, M. Morari, and M. N. Zeilinger, "Distributed synthesis and stability of cooperative distributed model predictive control for linear systems," *Automatica*, vol. 69, pp. 117–125, Jul. 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0005109816300413

[7] R. E. Jalal and B. P. Rasmussen, "Limited-Communication Distributed Model Predictive Control for Coupled and Constrained Subsystems," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 5, pp. 1807–1815, Sep. 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7707456/

[8] S. Adelipour, M. Haeri, and G. Pannocchia, "Decentralized Robust Model Predictive Control for Multi-Input Linear Systems," in *2018 UKACC 12th International Conference on Control (CONTROL)*. Sheffield: IEEE, Sep. 2018, pp. 13–18. [Online]. Available: https://ieeexplore.ieee.org/document/8516722/

[9] L. Furieri and M. Kamgarpour, "Robust control of constrained systems given an information structure," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. Melbourne, Australia: IEEE, Dec. 2017, pp. 3481–3486. [Online]. Available: http://ieeexplore.ieee.org/document/8264169/

[10] Z. Wang and C.-J. Ong, "Distributed MPC of constrained linear systems with online decoupling of the terminal constraint," in *2015 American Control Conference (ACC)*. Chicago, IL, USA: IEEE, Jul. 2015, pp. 2942–2947. [Online]. Available: http://ieeexplore.ieee.org/document/7171182/

[11] Y. Wang, N. Matni, and J. C. Doyle, "Separable and Localized System-Level Synthesis for Large-Scale Systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4234–4249, Dec. 2018.

[12] J. Anderson, J. C. Doyle, S. Low, and N. Matni, "System Level Synthesis," *arXiv:1904.01634 [cs, math]*, Apr. 2019, arXiv: 1904.01634. [Online]. Available: http://arxiv.org/abs/1904.01634

[13] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, Jun. 2017, google-Books-ID: 7NUoDwAAQBAJ.

[14] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523–533, Apr. 2006. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0005109806000021

[15] N. Matni, Y.-S. Wang, and J. Anderson, "Scalable system level synthesis for virtually localizable systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. Melbourne, Australia: IEEE, Dec. 2017, pp. 3473–3480. [Online]. Available: http://ieeexplore.ieee.org/document/8264168/

[16] M. A. Muller and F. Allgöwer, "Economic and Distributed Model Predictive Control: Recent Developments in Optimization-Based Control," *SICE Journal of Control, Measurement, and System Integration*, vol. 10, no. 2, pp. 39–52, 2017. [Online]. Available: https://www.jstage.jst.go.jp/article/jcmsi/10/2/10_39/_article

[17] S. Boyd, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010. [Online]. Available: http://www.nowpublishers.com/article/Details/MAL-016

[18] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004.

[19] G. Costantini, R. Rostami, and D. Gorges, "Decomposition Methods for Distributed Quadratic Programming with Application to Distributed Model Predictive Control," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. Monticello, IL, USA: IEEE, Oct. 2018, pp. 943–950. [Online]. Available: https://ieeexplore.ieee.org/document/8636067/

[20] "Gurobi Optimizer Reference Manual," p. 693.

[21] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.0 beta." Sep. 2013. [Online]. Available: http://cvxr.com/cvx,September2013.[2]

[22] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*, ser. Lecture Notes in Control and Information Sciences. V. Blondel, S. Boyd, and H. Kimura, editors, Springer, 2008, pp. 95–110. [Online]. Available: http://stanford.edu/~boyd/graph_dcp.html

[23] C. Conte, T. Summers, M. N. Zeilinger, M. Morari, and C. N. Jones, "Computational aspects of distributed optimization in model predictive control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. Maui, HI, USA: IEEE, Dec. 2012, pp. 6819–6824. [Online]. Available: http://ieeexplore.ieee.org/document/6426138/