Automatica 93 (2018) 407–414

Contents lists available at ScienceDirect

Automatica

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

Brief paper

Scalable distributed model predictive control for constrained systems☆

Elham (Fatemeh) Asadi [a,*], Arthur Richards [b]

[a] Department of Engineering, University of Cambridge, Trumpington Street, Cambridge, UK
[b] Aerospace Engineering Department, Queens Building, University of Bristol, Bristol, UK

ARTICLE INFO

Article history:
Received 23 May 2016
Received in revised form 16 September 2017
Accepted 15 February 2018

Keywords:
Distributed model predictive control
Constrained systems
Scalable DMPC

ABSTRACT

A distributed model predictive control strategy is proposed for subsystems sharing a limited resource. Self-organized Time Division Multiple Access is used to coordinate subsystem controllers in a sequence such that no two re-optimize simultaneously. This new approach requires no central coordination or pre-organized optimizing sequence. The scheme guarantees satisfaction of coupled constraints despite dynamic entry and exit of subsystems.

1. Introduction

Controlling large-scale systems such as transport networks or power distribution grids in a centralized way is often hard due to the computational scaling and coordination requirements. Centralized control is also prone to single point of failure, motivating interest in distributed control systems. Model Predictive Control (MPC) is a control technique combining constrained optimization with feedback control (Grüne & Pannek, 2011; Maciejowski, 2002), and schemes for Distributed MPC (DMPC) have been discussed by Christofides, Scattolini, de la Peña, and Liu (2013), Negenborn and Maestre (2014), Scattolini (2009) and many more. This paper focusses on DMPC for subsystems sharing a limited resource, which couples the systems through constraints (Bourdais, Buisson, Dumur, Guéguen, & Moroşan, 2014; Keviczky, Borrelli, & Balas, 2006; Kuwata, Richards, Schouwenaars, & How, 2007a; Li, Shi, & Yan, 2016; Lucia, Kögel, & Findeisen, 2015; Müller, Reble, & Allgöwer, 2012; Tedesco, Raimondo, & Casavola, 2014). Other forms of coupling, not considered here, are through the dynamics (Alessio, Barcelli, & Bemporad, 2011; Dunbar, 2007; Farina & Scattolini, 2012; Hernandez & Trodden, 2016) or through the system-wide objective function (Borrelli & Keviczky, 2006; Dunbar & Murray, 2006; Wang & Ong, 2010).

This paper adopts a *serial* DMPC scheme in which only one subsystem controller may optimize its plan at a time. With the plans for other subsystems therefore fixed, and known via communication, system-wide feasibility is this ensured. Previous work on serial schemes has proven its properties, subject to the assumption of an agreed updating sequence for the subsystems (Dai, Xia, Gao, Kouvaritakis, & Cannon, 2015; Keviczky, Borrelli, & Balas, 2004a, 2004b; Kuwata et al., 2007a; Richards & How, 2007; Trodden & Richards, 2013). However, the determination of that sequence is a centralized process. The simple contribution of this paper is the incorporation of a distributed slot allocation process, inspired by multiple access channel (MAC) sharing methods from communications systems (Rom & Sidi, 2012). In particular, Self-organizing Time Division Multiple Access (STDMA) (Gaugel, Mittag, Hartenstein, Papanastasiou, & Strom, 2013) is adopted, but instead of allocating *transmission slots* for communication, here it allocates *optimization slots* for re-planning.

The goal of distributed sequencing (slot allocation) is the same in spirit with the goal of "Plug & Play (PnP) Control" which intends to handle the distributed control problem for systems with a changing numbers of subsystems. In PnP control by adding or removing a subsystem, just local controller of the subsystem under control and subsystems influenced by it (neighbours) need to be redesigned (Stoustrup, 2009). PnP control methods proposed by Riverso, Farina, and Ferrari-Trecate (2014) and Zeilinger, Pu, Riverso, Ferrari-Trecate, and Jones (2013) tackle the problem of automatically accommodating the constant changes in system model due to adding or removing one or multiple subsystems during closed-loop operation. In both of these works, subsystems

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Giancarlo Ferrari-Trecate under the direction of Editor Ian R. Petersen.

* Corresponding author.
E-mail addresses: e.asadi@eng.cam.ac.uk (E. Asadi), arthur.richards@bristol.ac.uk (A. Richards).

https://doi.org/10.1016/j.automatica.2018.03.050
0005-1098/

are physically coupled whereas in this paper subsystems are coupled through their constraints. Barreiro-Gomez, Obando, Ocampo-Martinez, and Quijano (2015) and Lucia et al. (2015) addressed the challenge of performing network changes because of joining and leaving subsystems with coupled constraints. The decentralized MPC scheme presented by Barreiro-Gomez et al. (2015) can handle only one single coupled constraint on control signals. The contract-based DMPC introduced by Lucia et al. (2015) guarantees the constraint satisfaction in parallel optimization via transmission of sequences of possible future trajectories. The proposed method in this paper considers the particular coupling constraints associated with sharing of limited resources. Also in contrast to Lucia et al. (2015), our subsystems communicate exact trajectories but with serial (one-at-a-time) optimization and they implement a decentralized approach to sequencing.

2. Self-organized sequencing

Consider a dynamic set of subsystems $\mathcal{P}(k)$ containing $n(k) = |\mathcal{P}(k)|$ members at each time step k. A subset of these subsystems $\mathcal{P}_C(k) \subseteq \mathcal{P}(k)$ is in the *cooperation mode*, using serial DMPC to coordinate their actions such that shared resource limits are respected. Let $n_C(k) = |\mathcal{P}_C(k)|$ denote the number of cooperating subsystems. The remainder $\mathcal{P}(k) \setminus \mathcal{P}_C(k)$ remain in a restricted *safe mode*, not consuming any of the shared resources, and hence not required to communicate. Serial DMPC requires a unique allocation of subsystems to time steps, such that every step is associated to at most one subsystem, $p_k \in \mathcal{P}_C(k) \cup \{0\}$. At every step, the allocated subsystem p_k (if there is one, $p_k \neq 0$) solves its local optimal control problem and shares the resulting intentions with the others. This section describes how the allocation is achieved in a dynamic, self-organized way, enabling subsystems to move from safe mode to cooperation mode. Since this problem is analogous to slot allocation in communications, the algorithm is based on STDMA, which is a decentralized MAC method.

Define $L_f \geq 1$ to be the *frame length*, *i.e.* the repeating period for slot allocation, such that $p_k = p_{k+L_f}$ provided p_k remains in cooperation. Then Algorithm 1 presents the procedure of self-organized sequencing for an agent $q \in \mathcal{P}(k) \setminus \mathcal{P}_C(k)$ wishing to enter cooperation mode. Since the allocation is periodic, entry involves simply listening for one frame and then choosing an available slot in the next frame. The possible problem is a "collision" in which two subsystems attempt to enter at the same step, each unaware of the presence of the other. This event is detected by both subsystems and a random back-off time is employed to avoid deadlock.

Leaving the cooperation is achieved by stopping transmission, indicating to others that the slot is again available. Thus, unlike the communications case where slot allocations have finite lifetime, a slot belongs to a subsystem indefinitely until that subsystem relinquishes it. The control constraints associated with entry and leaving are described in Section 4.

Assumption 1 (*Frame Length*)**.** The frame length L_f is known to all subsystem controllers. This forms part of the common interface for subsystems: it is central to the scalability concept that the interface is standard and known to all agents.

Remark 1. It is not necessary for all subsystem controllers to define a common phasing of the frames, since the frames are periodic (Rom & Sidi, 2012).

Remark 2. Since no more than L_f subsystems can have slots, then the frame length L_f represents a limit on the number of subsystems in cooperation mode: $n_C(k) \leq L_f$. The choice of L_f therefore represents an important design choice, as increasing L_f means more capacity for entering agents but a longer wait to enter, according to Algorithm 1. A full study of this trade-off is beyond the scope of this brief paper and the reader is directed to Asadi and Richards (2015) for more consideration.

Algorithm 1 Entry into Cooperation Mode

Require: Subsystem ID q, initial time k_1
1: Listen for L_f steps to determine $\{p_{k_1}, \ldots, p_{k_1+L_f}\}$
2: Identify offsets of free slots: $\mathcal{J}_{free} = \{j \in [0, \ldots, L_f] \mid p_{k_1+j} = 0\}$
3: **if** no free slot, $\mathcal{J}_{free} = \emptyset$ **then**
4: Try again: go to Step 1
5: **else**
6: Choose free slot at random, $\hat{j} \in \mathcal{J}_{free}$
7: Wait for slot $\hat{j}$ in next frame, $k = k_1 + L_f + \hat{j}$
8: Transmit current plan $Y_p^*(k_1 + L_f + \hat{j})$
9: **if** no other subsystem transmitted **then**
10: Secured $p_{k_1+\hat{j}+nL_f} = q \; \forall n = 1, 2, \ldots$ as long as $q \in \mathcal{P}_C(k)$
11: **return** Success
12: **else**
13: Collision: wait for random number of steps
14: Try again: go to Step 1
15: **end if**
16: **end if**

3. Control problem definition

Each subsystem $p \in \mathcal{P}(k)$ has its own dynamics,

$$\mathbf{x}_p(k+1) = f_p(\mathbf{x}_p(k), \mathbf{u}_p(k)) \quad k \in \mathbb{N}, \quad \forall p \in \mathcal{P} \tag{1}$$

where $\mathbf{x}_p \in \mathbb{R}^{N_x,p}$ and $\mathbf{u}_p \in \mathbb{R}^{N_u,p}$ are the state vector and control input vector of subsystem p, respectively.

Remark 3. The dynamics (1) are not subject to any uncertainty. Ideas such as the tube approach (Trodden & Richards, 2010) could be applied to handle disturbances, but these are omitted here for simplicity.

Each subsystem p is subject to local constraints on state and input

$$\mathbf{x}_p(k) \in \mathcal{X}_p \tag{2}$$

$$\mathbf{u}_p(k) \in \mathcal{U}_p \tag{3}$$

and has its own local objective function, in fixed horizon MPC form

$$J_p = \sum_{t=0}^{N-1} l_p(\mathbf{x}_p(k+t|k), \mathbf{u}_p(k+t|k), k) + V_p(\mathbf{x}_p(k+N|k), k) \tag{4}$$

where N is the number of time steps in the prediction horizon, $l_p : \mathbb{R}^{N_x,p} \times \mathbb{R}^{N_u,p} \times \mathbb{R} \to \mathbb{R}_{0+}$ represents a stage cost, V_p is a terminal cost and the double subscript notation $(k+t|k)$ indicates the prediction of a variable t steps ahead from time k. The focus of this paper is constraint satisfaction, and hence the nature of the cost function will not be specified in more detail. However, the reader should note that the time variation permits different costs to be used in different modes, and this will be exploited later.

Define a set of shared resources $\mathcal{L}$ and let $y_p^\ell \in \mathbb{R}$ be the amount of particular resource $\ell \in \mathcal{L}$ used by subsystem p:

$$y_p^\ell(k) = g_p^\ell(\mathbf{x}_p(k), \mathbf{u}_p(k)). \tag{5}$$

All subsystems share the limited resources $\mathcal{L}$ and hence their outputs are coupled by

$$\sum_{p=1}^{n} y_p^\ell(k) \leq 1, \; \forall \ell \in \mathcal{L}. \tag{6}$$