For Want of Parallelism in Distributed NMPC

Jürgen Pannek

Curtin University of Technology, Perth, 6845 WA, Australia

Abstract

We consider a distributed non cooperative control setting in which systems are interconnected via state constraints. Each of these systems is governed by an agent which is responsible for exchanging information with its neighbours and computing a feedback law using a nonlinear model predictive controller to avoid violations of constraints. For this setting we present an algorithm which allows us to run all agents mostly in parallel. Moreover, we show both feasibility and stability of the closed loop using only abstract properties of this algorithm. To this end, we utilize a trajectory based stability result which we extend to the distributed setting.

Keywords: nonlinear model predictive control, suboptimality, stability, parallel algorithm

1. Introduction

Distributed control problems can arise either naturally, i.e. by a set of coupled systems which shall be controlled, see, e.g., Dold and Stursberg [3], or if a large problem is decomposed into smaller, again coupled problems, see Rawlings and Mayne [15, Chapter 10] or Scattolini [18] for an overview. In the latter case, the general idea is that smaller problems are solvable easier and faster which allows to even overcompensate the computational effort to coordinate these systems, cf. Richards and How [17, Section 7]. In either case, one distinguishes between cooperative control which features a centralized objective, and non cooperative control where the objectives of the systems are independent from each other. Using a centralized objective there are several possibilities to divide the optimization problem...
into subproblems and if suitable conditions hold then similar performance of the distributed control obtained from these subproblems and the centralized control can be shown, see, e.g., Rawlings and Mayne [15, Chapter 10] or Giselsson and Rantzer [5].

Throughout this work we focus on the non cooperative control settings and for each system we impose an agent which exchanges state information with its neighbours and uses the local objective to compute a local control which satisfies the coupling constraints. In particular, for the computing task we focus on feedback design via nonlinear model predictive controllers (NMPC) which minimize the distance of the current state to the desired equilibrium over a finite time horizon. In order to show asymptotic stability of the closed loop, one often imposes additional stabilizing terminal constraints and costs, see, e.g., Keerthi and Gilbert [13] or Chen and Allgöwer [2] respectively. Since such terminal constraints may require long optimization horizons, we focus in the plain NMPC setting without those modifications. In the non distributed case, stability for such problems has been shown in Grüne et al. [10] whereas the distributed case is treated in Grüne and Worthmann [11] using the algorithm of Richards and How [16, 17].

Here, we prove an idea outlined in Grüne and Worthmann [11] based the trajectory based setting of Grüne and Pannek [9, Chapter 7] which allows us to significantly reduce the horizon length in the distributed case while maintaining sub-optimality estimates and stability like behavior of the closed loop. Since the computing time of the NMPC control law for each agent is not negligible, we present an algorithm which allows us to execute these computations mostly in parallel using priority and testing rules as well as a decision memory. From Rawlings and Mayne [15, Chapter 10] it is known that for the non cooperative control setting one can only expect to reach a Nash equilibrium. Yet, under suitable conditions the closed loop solutions may still be stable and maintain the coupling constraints. For the proposed algorithm we present conditions which guarantee feasibility of the closed loop and present necessary as well as sufficient conditions for asymptotic stability using only abstract properties of both the priority and the testing rule. While here we focus on the plain NMPC case, we also outline how feasibility and stability results can be obtained using NMPC with stabilizing terminal constraints or cost.

The paper is organized as follows: First, in Section 2 we formally define the problem under consideration for which we show different stability results for the distributed case in Section 3. In the central Section 4 we present a covering algorithm which allows us to run the computations of the agents mostly in parallel. Using this algorithm, we show necessary and sufficient conditions for stability of
the resulting closed loop and also how much parallelism can be achieved. Instead of a separated example section, we use an analytical example throughout the entire work to present the improvement of the stability result but also to illustrate the abstract functions used within the proposed algorithm in Section 4. Finally, we draw conclusions in Section 5 and present ideas for future research based on the presented work.

2. Setup and Preliminaries

Throughout this work we consider a set of nonlinear discrete time systems

\[ x_p(n + 1) = f_p(x_p(n), u_p(n)), \quad p \in \mathcal{P} := \{1, \ldots, P\}, n \in \mathbb{N}_0 \]  

(1)

with \( x_p(n) \in X_p \) and \( u_p(n) \in U_p \) and \( \mathbb{N}_0 \) denoting the set of natural numbers including zero. Here, \( X_p \) and \( U_p \), \( p \in \mathcal{P} \), are assumed to be arbitrary metric spaces denoting the state space and the set of admissible control values of the \( p \)-th system, respectively. The metrics to measure distances between two elements of \( X_p \) or of \( U_p \) are denoted by \( d_{X_p} : X_p \times X_p \to \mathbb{R}_{\geq 0} \) and \( d_{U_p} : U_p \times U_p \to \mathbb{R}_{\geq 0} \) where \( \mathbb{R}_{\geq 0} \) denotes the positive reals including zero. In the following we denote the solution of a system \( p \) of (1) corresponding to the initial value \( x_p(0) = x^0_p \) and the control sequence \( u_p(k) \in U_p \), \( k = 0, 1, 2, \ldots \), by \( x_p^u(k, x^0_p) \).

In order to define our goal we say that a continuous function \( \alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \) is of class \( \mathcal{K}_\infty \) if it satisfies \( \alpha(0) = 0 \), is strictly increasing and unbounded. A continuous function \( \gamma : \mathbb{R}_{\geq 0}^P \to \mathbb{R}_{\geq 0} \) is called a class \( \mathcal{K}_\infty^P \) function if it satisfies \( \gamma(0) = 0 \), is strictly increasing in each component and is unbounded. A continuous function \( \beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \) is of class \( \mathcal{KL} \) if it is strictly decreasing in its second argument with \( \lim_{t \to \infty} \beta(r, t) = 0 \) for each \( r > 0 \) and satisfies \( \beta(\cdot, t) \in \mathcal{K}_\infty \) for each \( t \geq 0 \). Moreover, \( \mathcal{B}_r(x) \) denotes the open ball with center \( x \) and radius \( r \) and for arbitrary \( x_1, x_2 \in X \) we denote the distance from \( x_1 \) to \( x_2 \) by \( \|x_1\|_{x_2} = d_X(x_1, x_2) \).

For the set of systems (1) the overall system is given by

\[ x(n + 1) = f(x(n), u(n)), \quad n \in \mathbb{N}_0 \]  

(2)

with state \( x(n) = (x_1(n)^\top, \ldots, x_P(n)^\top)^\top \in X = X_1 \times \ldots \times X_P \) and control \( u(n) = (u_1(n)^\top, \ldots, u_P(n)^\top)^\top \in U = U_1 \times \ldots \times U_P \). Now, our goal is to asymptotically stabilize system (2) at a desired equilibrium point \( x^{\text{ref}} \in X \), i.e. to fulfill the following:
Definition 1. Let $x^* \in X$ be an equilibrium for a system (2), i.e., there exists a $u \in U$ such that $f(x^*,u) = x^*$. Then we say that $x^*$ is \textit{locally asymptotically stable} if there exist $r > 0$ and a function $\beta \in KL$ such that the inequality

$$\|x(n,x^0)\|_{x_{\text{ref}}} \leq \beta(\|x^0\|_{x_{\text{ref}}}, n)$$

holds for all $x^0 \in B_r(x_{\text{ref}})$ and all $n \in \mathbb{N}_0$.

Here, we impose an agent for each system $p \in P$ which is used to both exchange information with other agents and to obtain suitable control sequences $u_p(\cdot) \in U_p^{\infty}$, that is controls satisfying $u_p(k) \in U_p$, $k = 0, 1, 2, \ldots$. Throughout this work each agent computes its control sequence via a nonlinear model predictive controller, a methodology which will be explained after Definition 2 below. In order to achieve asymptotic stability of the overall system (2) we develop a covering algorithm which allows us to run the NMPC computations (mostly) in parallel while maintaining possible state and control constraints. Throughout this work we incorporate such constraints by considering suitable subsets $X \subset X$, $U \subset U$ of the state and control value space for system (2).

Note that due to these constraints there possibly exist couplings between the systems $f_p$, $p \in P$. Since we want to compute local controls $u_p$ such that none of the constraints imposed by neighbouring systems are violated, we must define the local constraints for each single system $f_p$, $p \in P$. To this end, we “project” the constraint set $X$ to the state space of a subset of systems.

Definition 2. For an index set $I_p = \{p_1, \ldots, p_m\} \subset P$ with $m \in \mathbb{N}$, $m \leq P$ and $p_i \neq p_j$ for all $i, j \in \{1, \ldots, m\}$ the set of partial states is defined as

$$X_{I_p} = X_{p_1} \times \ldots \times X_{p_m}$$

and we denote elements of $X_{I_p}$ by $x_{I_p} = (x_{p_1}, \ldots, x_{p_m})$. Accordingly, the partial state constraint set is defined by

$$\mathbb{X}_{I_p} := \{x_{I_p} \in X_{I_p} \mid \text{there is } \tilde{x} \in \mathbb{X} \text{ with } \tilde{x}_{p_i} = x \text{ for } i = 1, \ldots, m\}.$$
allow for changing network topologies, i.e. at time instants \( n \) and \( n + 1 \) we do not assume that the set of neighbours are identical. Additionally, we allow the case that even if neighbouring information of a system \( q \in \mathcal{P} \setminus \{p\} \) is known to an agent \( p \in \mathcal{P} \), agent \( p \) can still choose to ignore that information in the calculation of the control. Consequently, the communication graph and the dependency graph may differ. Moreover, we want to allow for considering older information about neighbours and variable lengths of this information. This leads us to the following definition of the exchanged neighbouring information:

**Definition 3.** Suppose that at time instant \( n \in \mathbb{N}_0 \) agent \( p \) knows the state sequences \( x_{q^n}^n(\cdot) = (x_{q^n}^0(0), \ldots, x_{q^n}^0(N_q)) \), \( N_q \in \mathbb{N}_0 \), computed at time instant \( n_q \leq n \) for a given neighbouring index set \( \mathcal{I}_p(n) \), that is \( q \in \mathcal{I}_p(n) \) with \( p \notin \mathcal{I}_p(n) \). We define the neighbouring information as

\[
\mathcal{I}_p(n) = \{(q, n_q, N_q, x_{q^n}^n(\cdot)) \mid q \in \mathcal{I}_p(n)\}
\]

being an element of the set \( \mathbb{I}_p = 2^Q \) with \( Q = (\mathcal{P} \setminus \{p\}) \times \mathbb{N}_0 \times \mathbb{N} \times X^\mathbb{N} \).

Knowing the states of neighbouring systems for a certain time period, we can define the index set used within the “projection” of the constraint set \( \mathbb{X} \).

**Definition 4.** For a given time instant \( n \in \mathbb{N}_0 \) and an agent \( p \in \mathcal{P} \) with neighbouring information \( \mathcal{I}_p(n) \), we call the set of systems \( q \in \mathcal{I}_p(n) \setminus \{p\} \) which are imposing constraints on system \( p \) at time instant \( n + k \in \mathbb{N}_0 \), \( k \geq 0 \) neighbouring prediction index set. This set is given by

\[
\mathcal{I}_p(n, k) = \{q \in \mathcal{I}_p(n) \setminus \{p\} \mid n + k \leq n_q + N_q\}.
\]

Now that we have defined the partial state constraint set connected to neighbouring information which is available to an agent, we can define the set of admissible controls from which the control sequence \( u_p(\cdot) \) can be chosen:

**Definition 5.** Given a time instant \( n \in \mathbb{N}_0 \) and an agent \( p \in \mathcal{P} \) with initial value \( x_p^0 \) and neighbouring information \( \mathcal{I}_p(n) \), we define the **set of admissible control sequences** for system \( p \) at time instant \( n \) as

\[
\mathbb{U}_p^{ad}(n, x_p^0, \mathcal{I}_p(n)) = \{u_p(\cdot) \in U_p^{\mathbb{N}_0} \mid \text{for all } k = 0, 1, \ldots \text{ we have } u_p(k) \in U_p \text{ and } (x_p^0(k, x_p), (x_q^0(k + n - n_q))_{\mathcal{I}_p(n, k)}) \in \mathbb{X}_{\{p\} \cup \mathcal{I}_p(n, k)}\}.
\]
Using an NMPC algorithm is one possibility to compute a control from the set of admissible controls. In particular, the method tries to approximate a control sequence such that the functional

\[
J^\infty_p(x_0^p, u_p) = \sum_{k=0}^{\infty} \ell_p(x_0^p(k), x_0^p(k), u_p(k))
\]

is minimized over all admissible control sequences, that is sequences \( u_p(\cdot) \) with \( u_p(k) = u_p^*(0) \) for all \( k \in \mathbb{N}_0 \) with \( u_p^* \in \mathbb{U}_p^{ad}(k, x_0^p(k), I_p(k)) \). Here, the function \( \ell_p \) is a stage cost function penalizing both the distance of the state to the desired equilibrium and the used control. A popular choice for this function is

\[
\ell_p(x^p, u^p) = \|x_p\|_{x_{ref}^p} + \lambda \|u_p\|_{u_{ref}^p}
\]

with weighting parameter \( \lambda > 0 \).

Since computing a control minimizing (4) is in general computationally intractable, the NMPC algorithm uses the truncated cost functional

\[
J^{N_p}_p(x_0^p, u_p) = \sum_{k=0}^{N_p-1} \ell_p(x_0^p(k), x_0^p(k), u_p(k))
\]

with finite horizon of length \( N_p \) and initial value \( x_0^p \) and computes a finite minimizing control sequence \( u^*_p \in U_p^{N_p,ad}(n, x_0^p(n), I_p(n)) \) with

\[
U_p^{N_p,ad}(n, x_0^p(n), I_p(n)) = \{ u_p(\cdot) \in U_p^{N_p} \mid \text{for all } k = 0, \ldots, N_p \text{ we have } u_p(k) \in U_p \text{ and } (x_0^p(k), x_0^p(k), (x_0^p(k+n-n_q))_{I_p(n,k)}) \in X(p) \cup I_p(n,k) \}.
\]

In the following we assume that a minimizing control sequence exists and denote the corresponding optimal value function by

\[
V^{N_p}_p(x_p(n), I_p(n)) = \min_{u_p \in U_p^{N_p,ad}(n, x_p(n), I_p(n))} J^{N_p}_p(x_p(n), u_p)
\]

where the minimizing control sequence is given by

\[
u_p^* = \arg\min_{u_p \in U_p^{N_p,ad}(n, x_p(n), I_p(n))} J^{N_p}_p(x_p(n), u_p).
\]

Here we use the \( \arg\min \) operator in the following sense: for a map \( a : U \to \mathbb{R} \), a nonempty subset \( \bar{U} \subseteq U \) and a value \( u^* \in \bar{U} \) we write

\[
u^* = \arg\min_{u \in \bar{U}} a(u)
\]
if and only if \( a(u^*) = \inf_{u \in U} a(u) \) holds. Whenever (6) holds the existence of the minimum \( \min_{u \in U} a(u) \) follows. However, we do not require uniqueness of the minimizer \( u^* \). In case of uniqueness equation (6) can be understood as an assignment, otherwise it is just a convenient way of writing “\( u \) minimizes \( a(u) \).”

Havin obtained a minimizing sequence \( u^*_p(\cdot) \), only the first element \( u^*_p(0) \) of the control sequence is implemented and the entire problem is shifted forward in time by one time instant and both a new initial value and neighbouring information need to be obtained. Applying this method iteratively results in a feedback law which assigns the first element of the minimizing control sequence \( u^*_p(\cdot) \) to the current state of the \( p \)-th system \( x_p(n) \) and the neighbouring information \( I_p(n) \) of the corresponding agent, i.e. a map

\[
\mu^N_p : (x_p(n), I_p(n)) \mapsto u^*_p(0).
\] (7)

Accordingly, the closed loop solution of the \( p \)-th system is given by

\[
x_p(n + 1) = f(x_p(n), \mu^N_p(x_p(n), I_p(n))) \quad \text{with} \quad x_p(0) = x_0^p
\] (8)

Now we have to ask: Can we compute such feedbacks in parallel? And if not, how much parallelism is possible? But probably the most important question is: Under which conditions is asymptotic stability still possible if we work mostly in parallel? Here, we start off by giving a trajectory based stability condition for the distributed NMPC case which we will use in Section 4 to answer the last question.

3. Stability

While commonly endpoint constraints or a Lyapunov function type endpoint weight are used to ensure stability of the closed loop, see, e.g., the articles of Keerthi and Gilbert [13], Chen and Allgöwer [2], Jadbabaie and Hauser [12] and Graichen and Kugi [6], we consider the plain NMPC version without these modifications. In order to guarantee stability in this case, we use the “relaxed” version of the dynamic programming principle, cf. Lincoln and Rantzer [14]. In particular, one can show asymptotic stability of (2) in a trajectory based setting using a relaxed Lyapunov condition as shown in Grüne and Pannek [9, Proposition 7.6]:

**Proposition 6.** Consider the feedback law \( \mu^N : \mathbb{X} \to \mathbb{U} \) and the closed loop trajectories \( x(\cdot) \) of (2) with control \( u = \mu^N \) and initial values \( x(0) \in \mathbb{X} \) to be given. If the optimal value function \( V^N : \mathbb{X} \to \mathbb{R}_{\geq 0} \) satisfies

\[
V^N(x(n)) \geq V^N(f(x(n), \mu^N(x(n)))) + \alpha \ell(x(n), \mu^N(x(n)))
\] (9)
for some $\alpha \in (0, 1]$ and all $n \in \mathbb{N}_0$, then
\begin{equation}
\alpha V^\infty(x(n)) \leq \alpha J^\infty(x(n), \mu^N) \leq V^N(x(n)) \leq V^\infty(x(n)) \tag{10}
\end{equation}
holds for all $n \in \mathbb{N}_0$.

If, in addition, there exist $\alpha_1, \alpha_2, \alpha_3 \in \mathcal{K}_\infty$ such that
\begin{equation}
\alpha_1(||x||_{\text{ref}}) \leq V^N(x) \leq \alpha_2(||x||_{\text{ref}}) \quad \text{and} \quad \ell(x, u) \geq \alpha_3(||x||_{\text{ref}}) \tag{11}
\end{equation}
holds for all $x(n) \in \mathbb{X}$ with $n \in \mathbb{N}_0$, then there exists a function $\beta \in \mathcal{KL}$ which only depends on $\alpha_1, \alpha_2, \alpha_3$ and $\alpha$ such that the inequality
\begin{equation}
||x(n)||_{\text{ref}} \leq \beta(||x(0)||_{\text{ref}}, n) \tag{12}
\end{equation}
holds for all $n \in \mathbb{N}_0$, i.e., $x$ behaves like a trajectory of an asymptotically stable system.

The key assumption in Proposition 6 is the relaxed Lyapunov–inequality (9) in which $\alpha$ can be interpreted as a lower bound for the rate of convergence. From the literature, it is well–known that this condition is satisfied for sufficiently long horizons $N$, cf. Jadbabaie and Hauser [12], Grimm et al. [7] or Alamir and Bornard [1], and that a suitable $N$ may be computed via methods described in Grüne and Pannek [9, Chapter 7] or Giselsson [4].

In a quite similar manner, an asymptotic stability result can be formulated in terms of the set of systems (1) instead of the combined system (2) if we assume the optimization horizons of each system $p \in \mathcal{P}$ to satisfy $N_p = N$:

**Proposition 7.** Consider the feedback laws $\mu^N_p : \mathbb{X}_p \times \mathbb{I}_p \to \mathbb{U}_p$ and the closed loop trajectories $x_p(\cdot)$ of (3) with initial values $x_p(0) \in \mathbb{X}_p$ to be given. If the optimal value functions $V^N_p : \mathbb{X}_p \to \mathbb{R}_{\geq 0}$ satisfy
\begin{equation}
V^N_p(x_p(n)) \geq V^N_p(f_p(x_p(n), \mu^N_p(x_p(n)), I_p(n)))) + \alpha \ell_p(x_p(n), \mu^N_p(x_p(n)), I_p(n))) \tag{13}
\end{equation}
for some $\alpha \in (0, 1]$ and all $n \in \mathbb{N}_0$, then for any weighting function $\gamma \in \mathcal{K}^\infty_0$ we have that (10) holds for all $n \in \mathbb{N}_0$ with
$V^N(x) := \gamma((V^N_1(x_1), \ldots, V^N_p(x_p))^\top)$ and $\ell(x, u) := \gamma((\ell_1(x_1, u_1), \ldots, \ell_p(x_p, u_p))^\top)$.

If, in addition, for every $p \in \mathcal{P}$ there exist $\alpha^p_1, \alpha^p_2, \alpha^p_3 \in \mathcal{K}_\infty$ such that
\begin{equation}
\alpha^p_1(||x_p||_{\text{ref}}) \leq V^N_p(x) \leq \alpha^p_2(||x_p||_{\text{ref}}) \quad \text{and} \quad \ell_p(x_p, u_p) \geq \alpha^p_3(||x_p||_{\text{ref}}) \tag{14}
\end{equation}
holds for all $x_p(n) \in \mathbb{X}$ with $n \in \mathbb{N}_0$, then there exists a function $\beta \in \mathcal{KL}$ which only depends on $\gamma, \alpha$ and all $\alpha^p_1, \alpha^p_2, \alpha^p_3$, $p \in \mathcal{P}$, such that (12) holds for all $n \in \mathbb{N}_0$.
Proof. Defining the abbreviations $V_p^N(x(n)) := (V_1^N(x_1(n)), \ldots, V_p^N(x_p(n)))^T$ and $\ell_p(x(n), \mu^N(x(n), I_p(n))) := (\ell_1(x_1(n), \mu_1^N(x_1(n), I_1(n))), \ldots, \ell_p(x_p(n), \mu_p^N(x_p(n), I_p(n))))^T$ we can combine all inequalities \((13)\) for $p \in P$ and obtain

$$\gamma(V_p^N(x(n))) \geq \gamma(V_p^N(x(n + 1))) + \alpha \gamma(\ell_p(x(n), \mu^N(x(n), I_p(n)))) \quad (15)$$

Now we can use the definition of $V_p^N$ and $\ell$ which gives us \((9)\). Hence, \((10)\) follows by definition of $V_p^N$ and $\ell$ which together with $\alpha_i(r) := \gamma((\alpha_1^i(r), \ldots, \alpha_3^i))$, $i = 1, 2, 3$, and again Proposition \(6\) shows the assertion. \hfill \Box

Certainly, condition \((13)\) would be desirable since it guarantees a decrease in $V_p^N$ for each $p \in P$. In practice, however, one would usually suspect $V_p^N$ to decrease for some $p \in P$ while it increases for others. Before we show a corresponding stability result, let us consider the following simple example:

**Example 8.** Consider the discrete systems

$$x_p(n + 1) = f_p(x_p(n), u_p(n)) = x_p(n) + u_p(n)$$

for $p = 1, 2$ with constraint sets

$$X = \{x \in \mathbb{Z}^4 \mid x_2 = 0 \text{ if } x_1 = 0 \text{ and } x_4 = 0 \text{ if } x_3 = 0 \text{ and } \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \neq \begin{pmatrix} x_3 \\ x_4 \end{pmatrix} \}$$

and $U = \{-1, 0, 1\}^4$. Suppose the running costs are given by

$$\ell_p(x_p, u_p) = \|x_p - x_p^{\text{ref}}\|_2^2$$

with $x_p^{\text{ref}} = (2, 0)^T$ and $x_p^{\text{ref}} = (-2, 0)^T$. Considering the initial value $x = (-1, 0, 1, 0)^T$ we now take a closer look at conditions \((13)\) and \((15)\):

Due to the constraint sets $X$, $U$ and the dynamics of the systems $f_p$, one of the agents $p$ has to move aside first to let the system of the other agent pass by before it can be steered towards its desired equilibrium. Without loss of generality we assume that system $p = 2$ moves aside. Since $V_2^N(x_2(0), I_2(0)) = V_2^N(x_2(1), I_2(1))$ holds for $N = 2$ and $N = 3$, we cannot guarantee \((13)\) to hold for these $N$. For larger values of $N$, however, we obtain

$$V_2^N(x_2(0), I_2(0)) = \begin{cases} 37 & \text{if } N = 4 \\
41 & \text{if } N = 5 \\
42 & \text{if } N \geq 6 \end{cases} \quad \text{and} \quad \ell_2(x_2(0), \mu_2^N(x_2(0), I_2(0))) = 9,$$
Proposition 9. Consider the feedback laws \( \mu^N_p : X_p \times U_p \to U_p \) and the closed loop trajectories \( x_p(\cdot) \) of (8) with initial values \( x_p(0) \in X_p \) to be given. If the optimal value functions \( V^N_p : X_p \to \mathbb{R}_+ \) satisfy (15) for some \( x \in \mathbb{R} \) and all \( n \in \mathbb{N}_0 \), then for any \( \gamma \in \mathcal{K}_{\infty}^p \) (10) holds for all \( n \in \mathbb{N}_0 \) with \( V^N \) and \( \ell \) defined as in Proposition 7.

If, in addition, for every \( p \in P \) there exist \( \alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty \) such that (14) holds for all \( x_p(n) \in X_p \) with \( n \in \mathbb{N}_0 \), then there exists a function \( \beta \in \mathcal{K} \) which only depends on \( \gamma, x \) and all \( \alpha_1^p, \alpha_2^p, \alpha_3^p, p \in P \) such that (12) holds for all \( n \in \mathbb{N}_0 \).

Proof. Follows directly from the proof of Proposition 7.
Example 10. Consider $\gamma$ to be the 1–norm, then we obtain for $N = 2$

$$
\sum_{p=1}^{2} V_p^N(x_p(0), I_p(0)) = 31 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(0), \mu_p^N(x_p(0), I_p(0))) = 18,
$$

$$
\sum_{p=1}^{2} V_p^N(x_p(1), I_p(1)) = 24 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(1), \mu_p^N(x_p(1), I_p(1))) = 13,
$$

$$
\sum_{p=1}^{2} V_p^N(x_p(2), I_p(2)) = 20 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(2), \mu_p^N(x_p(2), I_p(2))) = 11,
$$

$$
\sum_{p=1}^{2} V_p^N(x_p(3), I_p(3)) = 13 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(3), \mu_p^N(x_p(3), I_p(3))) = 9,
$$

$$
\sum_{p=1}^{2} V_p^N(x_p(4), I_p(4)) = 5 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(4), \mu_p^N(x_p(4), I_p(4))) = 4,
$$

$$
\sum_{p=1}^{2} V_p^N(x_p(5), I_p(5)) = 1 \quad \text{and} \quad \sum_{p=1}^{2} \ell_p(x_p(5), \mu_p^N(x_p(5), I_p(5))) = 1.
$$

and $\sum_{p=1}^{2} V_p^N(x_p(n), I_p(n)) = \sum_{p=1}^{2} \ell_p(x_p(n), \mu_p^N(x_p(n), I_p(n))) = 0$ for $n \geq 6$. Hence, (15) holds with $\alpha = 4/13$ and we obtain asymptotic stability of the closed loop by Proposition 9. Since $\alpha > 0$ holds for all $N \geq 2$ this nicely illustrates the advantage of considering condition (15) instead of (13).

As outlined before Proposition 9, under certain conditions the algorithm of Richards and How [16, 17] can be applied to generate solutions such that (13) and (14) hold. However, the nature of this algorithm is sequential, that is while one agent $p \in \mathcal{P}$ is computing its control, all other agents $q \in \mathcal{P} \setminus \{p\}$ have to wait until agent $p$ finished computing. Hence, if the number of systems $P$ is large, such an algorithm may cause rather long waiting times, a feature which may be unwanted if fast sampling is used. Still, as noted in Richards and How [17, Section 7], due to its decentralized nature the dimension of each problem is significantly smaller and hence the algorithm reduces the numerical effort compared to a centralized solution considerably. Apart from the sequential nature, the algorithm of Richards and How requires accessibility to the full neighbouring information, i.e. a full communication graph, and an agent $p \in \mathcal{P}$ always uses the latest available neighbouring information.
to compute a minimizing control $u^*_p$ which results in a full dependency graph. While the latter condition on the dependency graph may be relaxed easily, it is a complex task to obtain a parallel algorithm and to relax the requirement of a full communication graph.

4. The covering algorithm

In this section we provide a covering algorithm which allows us to run the agents $p \in \mathcal{P}$ mostly in parallel. Unfortunately, working in a parallel distributed setting omits the use of standard techniques from optimization such as first and second order information of the cost functional and the constraints for the interlink between systems to search for optimal controls.

Here, we present a very general algorithm to circumvent this deficiency using the abstract maps $\Pi, \Theta : 2^\mathcal{P} \to 2^\mathcal{P}$ which denote priority and testing rules. The general idea of the algorithm is to first generate priority lists of the systems according to the rule $\Pi$, which is actually a permutation map, and according to their interconnection with other systems – just as the right-before-left rule in street traffic or the search direction in optimization methods. Secondly, these lists are used to remember earlier decisions which avoids generating periodic behaviour. This part of the algorithm is inspired by Bland’s rule and the lexicographic ordering method used in the simplex algorithm to cope with degeneracy. Last, the testing rule $\Theta$, which is actually a self-concatenated mapping, offers a possibility to break up earlier decisions which avoids blockages and reduces both the number of priority lists and thereby the numerical effort to compute the control sequences.

The structural layout of the algorithm we present now is closely related to the ordinary NMPC algorithm outlined in Section 2:

**Algorithm 11.** Set lists $\mathcal{P}_1 := (1, \ldots, P)$ and $\mathcal{P}_p := \emptyset$ for $p = 2, \ldots, P$, $n := 0$ and $I_p(n) := \emptyset$ for $p = 1, \ldots, P$.

1. Obtain new measurements $x_p(n)$ for $p \in \mathcal{P}$.

2a. For $i$ from 2 to $P$ do
   For $j$ from 1 to $\#\mathcal{P}_i$ do
     (i) Set $I_{ij}(n) := \Theta(I_{ij}(n)) \subsetneq I_{ij}(n)$
     (ii) If $I_{ij}(n) = \emptyset$, then remove $i_j$ from $\mathcal{P}_i$ and set $\mathcal{P}_1 := (\mathcal{P}_1, i_j)$
     Else: If $\tilde{m} = \min_{k \in \mathcal{P}_m, p_k \in I_{ij}(n)} m < i$ holds, then remove $i_j$ from $\mathcal{P}_i$ and set $\mathcal{P}_{\tilde{m}} := (\mathcal{P}_{\tilde{m}}, i_j)$
2b. Compute a control \( u_p^*(\cdot) \) minimizing (4) or (5) with \( x_p^0 = x_p(n) \) for \( p \in P \) in parallel and send information to all agents \( q \in \{ q \in P \mid q \in P_j, p \in P_i \text{ and } j \geq i \} \)

2c. For \( i \) from 1 to \( P \) do
   (i) If \( \#P_i \in \{ 0, 1 \} \), goto Step 3.
   Else: Sort index list by setting \( P_i := \Pi(P_i) \)
   (ii) For \( j \) from 2 to \( \#P_i \) do
      If system \( p_{ij} \) violates constraints imposed by systems \( p_{ik}, k < j \), then set \( P_{i+1} := (P_{i+1}, i, j) \) and \( I_{ij}(n) := I_{ij}(n) \cup \{ p_{ik} \in P_1 \setminus P_{i+1} \mid p_{ik}, k < j, \text{ induces constraints violated by system } p_{ij} \} \)
   (iii) Set \( P_i := P_i \setminus P_{i+1} \)
   (iv) Compute a control \( u_p^*(\cdot, I_p(n)) \) minimizing (4) or (5) for all \( p \in P_{i+1} \) in parallel and send information to all agents \( q \in \{ q \in \mathcal{P} \mid q \in P_j, j \geq i \} \)

3. Implement \( \mu_p^{N_p}(x_p(n), I_p(n)) := u_p^*(0) \), set \( n := n + 1 \) and goto Step 1.

Before answering the question how much parallelism is possible by taking a closer look at the rules \( \Pi, \Theta \) and the properties of Algorithm 11, we first consider the question whether a feasible feedback \( \mu_p^{N_p} \) can be computed via Algorithm 11:

**Theorem 12.** Assume a feasible initial value \( x_0 \in \mathbb{X} \) for system (2) to be given. Suppose that for all \( p \in \mathcal{P} \) and all \( n \in \mathbb{N}_0 \) we have that the sets of admissible controls \( \mathcal{U}_p^\text{ad}(n, x_p(n), I_p(n)) \) in case of cost functional (4) or \( \mathcal{U}_{p}^{N_p, \text{ad}}(n, x_p(n), I_p(n)) \) in case of cost functional (5) in Steps 2b and 2c(iv) are not empty, then the closed loop solutions (3) satisfy \( x(n) = (x_1(n)^T, \ldots, x_p(n)^T)^T \in \mathbb{X}. \)

**Proof.** Using \( x_0 \in \mathbb{X} \) and \( \mathcal{U}_p^\text{ad}(0, x_p(0), I_p(0)) \neq \emptyset \) for all \( p \in \mathcal{P} \) in case of cost functional (4) or \( \mathcal{U}_{p}^{N_p, \text{ad}}(0, x_p(0), I_p(0)) \neq \emptyset \) for all \( p \in \mathcal{P} \) in case of cost functional (5), we obtain from Steps 2b and 2c(iv) that optimal controls \( u_p^*(\cdot, I_p(0)) \) exist for all \( p \in \mathcal{P} \). Hence, by definition of the closed loop in (3) and Step 3 we obtain that \( x(1) = (x_1(1)^T, \ldots, x_p(1)^T)^T \in \mathbb{X} \) holds. Applying the same argumentation inductively for all \( n \in \mathbb{N}_0 \) the assertion follows.

The idea of the priority rule is straightforward. In particular, we have already used it in Example 8 to solve the blockage in the very first step:

**Example 13.** Again consider Example 8 as already mentioned, due to the constraint sets \( \mathbb{X}, \mathbb{U} \) and the dynamics of the systems \( f_p \), one of the agents \( p \) has to move aside first to let the system of the other agent pass by before it can be steered.
towards its desired equilibrium. Putting priority of agent $p = 1$ into a mathematical form, we see that $\Pi$ is a lexicographic ordering, that is a list $\mathcal{L}$ is mapped to its minimal permutation with respect to the dictionary ordering $<^d$ induced by the total orderings $\{<_1, \ldots , <_m\}$ where $m$ is the length of the list $\mathcal{L}$ and $<_i, i = 1, \ldots m$ is the usual ordering $<$ of the natural numbers $\mathbb{N}$.

Apart from the lexicographic ordering, also other heuristics like the greedy heuristic might be used to sort a list of systems according to their impact on the stage cost $\ell_p$. Different from the lexicographic ordering in this case the total orderings $<_i, i = 1, \ldots , m$ where $m$ is again the length of a list $\mathcal{L}$ are given by

$$p_1 <_i p_2 \text{ if } \ell_{p_1}(x_{p_1}(n), u_{p_1}(n), I_{p_1}(n)) < \ell_{p_2}(x_{p_2}(n), u_{p_2}(n), I_{p_2}(n)).$$

Yet, it is not clear how the priority rule should be chosen in a nonlinear setting, and throughout this work we will not focus on this question but instead concentrate on general properties of Algorithm \[11\].

The idea of the testing rule $\Theta$ is more involved as it may interfere with the idea of keeping track of earlier decisions. As mentioned before, its purpose is to reduce the number of the priority lists and the numerical effort to compute the control sequences $u^*_p$. The reason why we want to reduce the number of priority lists is that Step 2c of Algorithm \[11\] is a sequential call for all lists $\mathcal{P}_i$. Accordingly, agents $p \in \mathcal{P}_{i+1}$ always have to wait until all agents $p \in \mathcal{P}_i$ have finished computing, a fact we wish to avoid. Note that this sequential nature is independent from the parallel computation of control sequences $u^*_p$, $p \in \mathcal{P}_i$. Using the testing rule $\Theta$ allows us to “test” whether a system $p \in \mathcal{P}_i$ still interferes with all systems $p \in \mathcal{P}_k$, $k < i$, or if it can be inserted into a different priority list $\mathcal{P}_k$, $k < i$, causing the number of lists and hence the number of non parallel steps to reduce. Yet even if system $p$ cannot be inserted in a different priority list, applying the testing rule might still result in reducing the size of the neighbouring index set $I_p(n)$. If this is the case, then the number of constraints of system $p$ is reduced which in turn reduces the numerical effort to compute the control sequence $u^*_p$.

**Example 14.** Consider once more Example \[8\] we see that we can use the testing rule $\Theta$ which always maps to the empty set. Applying Algorithm \[11\] we therefore obtain that only for $n = 0$ and $n = 1$ we have that $u^*_2$ depends on the solution of system $p = 1$ whereas for all other time instants both problems can be solved independently from each other.

Still, even if we do not know the exact sorting and testing operators $\Pi, \Theta$, we can still tackle the question on how much parallelism is possible by using
conditions on the priority lists \( \mathcal{P}_i \) generated in Algorithm 11. Our first result actually shows in which case all agents can compute their controls independently from each other:

**Lemma 15.** Suppose that for given systems (1), maps \( \Pi, \Theta : 2^{\mathbb{P}} \to 2^{\mathbb{P}} \) and \( n \in \mathbb{N}_0 \) we have that \( \mathcal{P}_2 = \emptyset \) holds in Step 2c(i) of Algorithm 11. Then every agent \( p \in \mathcal{P} \) can compute its control sequence independently of all other agents \( q \in \mathcal{P} \setminus \{ p \} \).

**Proof.** Since \( \mathcal{P}_2 = \emptyset \) Step 2c(ii) of Algorithm 11 guarantees that there are no systems \( p_1, p_2 \in \mathcal{P}_1, p_1 \neq p_2 \), such that \( p_1 \) induces a constraint on \( p_2 \) which is violated by \( p_2 \), i.e. \( I_{p_1}(n) = \emptyset \) for all \( p \in \mathcal{P} \). Hence, for each agent \( p \in \mathcal{P} \) the set of admissible controls simplifies to

\[
\mathcal{U}^{\text{ad}}_p(n, x_0^p, I_p(n)) = \{ u_p(\cdot) \in U_p^{\mathbb{N}_0} | u_p(k) \in U_p \text{ and } x^0_p(k, x_0^p) \in X_p \text{ for all } k \in \mathbb{N}_0 \}
\]

if cost functional (4) or

\[
\mathcal{U}^{N, \text{ad}}_p(n, x_0^p, I_p(n)) = \{ u_p(\cdot) \in U_p^{N_p} | u_p(k) \in U_p \text{ and } x^0_p(k, x_0^p) \in X_p \text{ for all } k \in \{0, \ldots, N_p\} \}
\]

if cost functional (5) is considered with \( x_0^p = x_p(n) \) showing the assertion. \( \square \)

Using the self-concatenation property of the map \( \Theta \), we can also show that under certain conditions the priority lists show dependency of agents:

**Lemma 16.** Consider systems (1), \( P \geq 2 \) to be given. Suppose that applying Algorithm 11 for given maps \( \Pi, \Theta : 2^{\mathbb{P}} \to 2^{\mathbb{P}} \) we have that \( \mathcal{P}_i \neq \emptyset \) with \( i \geq 2 \) holds for some \( n \geq \pi \) and all \( \pi \in \mathbb{N}_0 \). Then for each system \( p \in \mathcal{P}_i \) there exists at least one system \( q \in \mathcal{P}_j, j < i \) such that \( q \in I_p(n) \). Moreover, in case cost functional (4) is used, we have

\[
u^*_p = \arg \min_{u_p \in \mathcal{U}^{\text{ad}}_p(n, x_0^p, \emptyset)} J^\infty_p(x_0^p(n), u_p) \notin \mathcal{U}^{\text{ad}}_p(n, x_0^p(n), I_p(n)) \subseteq \mathcal{U}^{\text{ad}}_p(n, x_0^p(n), I_p(n))
\]

and in case of cost functional (5) we have

\[
u^*_p = \arg \min_{u_p \in \mathcal{U}^{N, \text{ad}}_p(n, x_0^p, \emptyset)} J^{N_p}_p(x_0^p(n), u_p) \notin \mathcal{U}^{\text{ad}}_p(n, x_0^p(n), I_p(n)) \subseteq \mathcal{U}^{N, \text{ad}}_p(n, x_0^p(n), I_p(n)) \subseteq \mathcal{U}^{N, \text{ad}}_p(n, x_0^p(n), \emptyset).
\]
Proof. Suppose that \( \mathcal{P}_i \neq \emptyset \) with \( i \geq 2 \) holds for some \( n \geq \overline{n} \) and all \( \overline{n} \in \mathbb{N}_0 \) and fix \( p \in \mathcal{P}_i \) arbitrarily. Suppose furthermore that there exists no \( q \in \mathcal{P}_j, j < i \) such that \( q \in \mathcal{I}_p(n) \) holds. Then, by the testing rule \( \Theta \) and Step 2c(ii) we obtain that there exists \( \overline{n} \in \mathbb{N}_0 \) such that \( \mathcal{P}_i = \emptyset \) for all \( n \geq \overline{n} \) contradicting our assumption. Hence, since \( p \in \mathcal{P}_i \) was chosen arbitrarily, we obtain that for each \( p \in \mathcal{P}_i \) there exists a system \( q \in \mathcal{I}_p(n) \), \( j < i \) such that \( q \in \mathcal{I}_p(n) \) holds.

Now, due to Step 2c(ii) and the fact that there exists a system \( q \in \mathcal{I}_p(n) \) imposing constraints on system \( p \) which are violated if \( q \notin \mathcal{I}_p(n) \) the assertion for both cost functionals (4) and (5) follows.

Now we can use Lemma 16 to answer the question under which conditions asymptotic stability can be shown. In particular, we first prove a necessary condition for asymptotic stability of (2).

**Theorem 17.** Consider systems (1), \( P \geq 2 \) to be given. Suppose that applying Algorithm 11 for all maps \( \Pi, \Theta : 2^P \rightarrow 2^P \) we have that \( \mathcal{P}_i \neq \emptyset, i \geq 2 \) holds for some \( n \geq \overline{n} \) and all \( \overline{n} \in \mathbb{N}_0 \) with \( \mathcal{I}_p(n, 1) \neq \emptyset \) for some \( p \in \mathcal{P}_i \). Then there exists no function \( \beta \in \mathcal{KL} \) such that (12) holds for all \( n \in \mathbb{N}_0 \).

**Proof.** Fix maps \( \Pi, \Theta : 2^P \rightarrow 2^P \). Then Lemma 16 states that for each system \( p \in \mathcal{P}_i \), there exists a system \( q \in \mathcal{P}_j, j < i \) such that \( q \notin \mathcal{I}_p(n) \). If for any \( p \in \mathcal{P}_i \) and any \( n \in \mathbb{N}_0 \) we have that \( \mathcal{U}_{p, \text{ad}}(n, x_p(n), I_p(n)) = \emptyset \) or \( \mathcal{U}_{p, \text{ad}}^N(n, x_p(n), I_p(n)) = \emptyset \) in case if cost functional (4) or (5) are used, respectively, we are done since no admissible solution exists. Otherwise, we obtain \( u_p^*(\cdot) \neq u_p^*(-) \) with

\[
\begin{align*}
u_p^1(\cdot) &= \arg\min_{u_p \in \mathcal{U}_{p, \text{ad}}(n, x_p(n), I_p(n))} J_p^\infty(x_p(n), u_p), \\
u_p^2(\cdot) &= \arg\min_{u_p \in \mathcal{U}_{p, \text{ad}}^N(n, x_p(n), I_p(n))} J_p^\infty(x_p(n), u_p)
\end{align*}
\]

in case of cost functional (4) and with

\[
\begin{align*}
u_p^1(\cdot) &= \arg\min_{u_p \in \mathcal{U}_{p, \text{ad}}^N(n, x_p(n), I_p(n))} J_p^N(x_p(n), u_p), \\
u_p^2(\cdot) &= \arg\min_{u_p \in \mathcal{U}_{p, \text{ad}}^N(n, x_p(n), I_p(n))} J_p^N(x_p(n), u_p)
\end{align*}
\]

in case of cost functional (5).

Hence, due to the fact that \( x_p^*(k, x(n)) \) for some \( k \) violates a constraint imposed by system \( q \) which is not violated by \( x_p^*(k, x(n)) \), we obtain that the open loop trajectories \( x_p^1(\cdot, x(n)) \) and \( x_p^2(\cdot, x(n)) \) differ. Using \( \mathcal{I}_p(n, 1) \neq \emptyset \), we can conclude that there exists a \( \delta_1 > 0 \) such that \( d_X(x_p^1(1, x_p(n)), x_p^2(1, x_p(n))) > \delta_1 \) holds. Since we always implement the first element of each optimal admissible
control, we have that \(d_X(f_p(x_p(n), u_p^1(0)), f_p(x_p(n), u_p^2(0))) > \delta_1\) holds. Now we have to consider two cases: If \(x_p(n+1) = x_p^{\text{ref}}\), then we can use the fact that the deviation \(d_X(x_p^{u_1^p}(1, x_p(n)), x_p^{u_2^p}(1, x_p(n))) > \delta_1\) will occur again for some \(\hat{n} > n\) due to the assumptions of the theorem. If \(x_p(n+1) \neq x_p^{\text{ref}}\), we immediately obtain the existence of a \(\delta_2 > 0\) such that \(\|x_p(n+1)\|_{\text{ref}} > \delta_2\) holds. In either case, we obtain that there exists a time index \(\hat{n} > n\) such that \(\|x_p(\hat{n})\|_{\text{ref}} > \delta = \min(\delta_1/2, \delta_2)\) holds.

Now suppose there exists a function \(\beta \in \mathcal{KL}\) such that (12) holds for all \(n \in \mathbb{N}_0\). Due to the \(L\)-property \(\beta\) in its second argument, we have that for each \(\varepsilon > 0\) there exists a \(\hat{n} \in \mathbb{N}_0\) such that \(\|x(n)\|_{\text{ref}} < \varepsilon\) for all \(n \geq \hat{n}\). Now we choose \(\varepsilon < \delta\) and \(\hat{n} \in \mathbb{N}_0\) accordingly. Since Lemma [16] holds for all \(\overline{n} \in \mathbb{N}_0\), we can conclude that for \(\hat{n} > n \geq \overline{n} = \hat{n}\) the inequality \(\|x(\hat{n})\|_{\text{ref}} \geq \|x_p(\hat{n})\|_{\text{ref}} > \delta > \varepsilon\) holds. This contradicts the existence of a function \(\beta \in \mathcal{KL}\) such that (12) holds for all \(n \in \mathbb{N}_0\). Last, since the maps \(\Pi\) and \(\Theta\) were chosen arbitrarily, the argumentation holds for all choices of \(\Pi\) and \(\Theta\) which completes the proof.

**Remark 18.** Condition \(I_p(n, 1) \neq \emptyset\) in Theorem [17] is required since from \(q \in I_p(n)\) we can only conclude that \(x_p^{u_1^p}(k_n, x_p(n))\) and \(x_p^{u_2^p}(k_n, x_p(n))\) differ for some \(k_n \geq 0\). Now, according to the NMPC algorithm, only the first control element is implemented and we may face the situation that again \(x_p^{u_1^p}(k_{n+1}, x_p(n + 1))\) and \(x_p^{u_2^p}(k_{n+1}, x_p(n + 1))\) differ for some \(k_{n+1} \geq k_n\). Now if \(k_n > 0\) holds for all \(n \in \mathbb{N}_0\), then system \(p\) may be asymptotically stable.

Unfortunately, the converse of Theorem [17] does not hold, not even in the special case that the conditions of Lemma [15] hold for all \(n \geq \overline{n}\) with \(\overline{n} \in \mathbb{N}_0\). This conclusion is due to the fact that even if \(\mathcal{P}_2 = \emptyset\) we can only guarantee that a control which minimizes (5) for all systems \(p \in \mathcal{P}\) can be computed without having to consider any other system \(q \in \mathcal{P} \setminus \{p\}\), but not whether all systems are actually stable.

**Theorem 19.** Suppose that for given maps \(\Pi, \Theta : 2^P \to 2^P\) we have that for a given initial value \(x_0 \in X\) the set of admissible controls \(\bigcup_{p \in \mathcal{P}, ad}^N_n(x_p(n), I_p(n))\) is not empty for all \(p \in \mathcal{P}\) and all \(n \in \mathbb{N}_0\). Suppose furthermore that there exist \(\alpha_1^p, \alpha_2^p, \alpha_3^p \in \mathcal{K}_\infty\), \(\gamma \in \mathcal{K}_\infty\) and \(\alpha > 0\) such that inequalities (14) and (15) hold for all \(n \in \mathbb{N}_0\). Then there exists a function \(\beta \in \mathcal{KL}\) which only depends on \(\alpha, \gamma\) and all \(\alpha_1^p, \alpha_2^p, \alpha_3^p, p \in \mathcal{P}\), such that (12) holds for all \(n \in \mathbb{N}_0\).

Moreover, there exists an \(\overline{n} \in \mathbb{N}_0\) such that for each \(n \geq \overline{n}\) we either have that \(\mathcal{P}_i \neq \emptyset\), \(i \geq 2\) holds with \(I_p(n, 1) = \emptyset\) for all \(p \in \mathcal{P}_i\) or \(\mathcal{P}_2 = \emptyset\).
Proof. Using \( x_0 \in X, \bigcup_{P}^{N} P_{\text{ad}}(n, x_p(n), I_p(n)) \neq \emptyset \) for all \( p \in P \) and all \( n \in \mathbb{N}_0 \) and Theorem 12 we obtain that the closed loop solution \( x(n) = (x_1(n)^T, \ldots, x_P(n)^T)^T \) exists for all \( n \in \mathbb{N}_0 \) and satisfies \( x(n) \in X \) for all \( n \in \mathbb{N}_0 \). Now, since inequalities (14) and (15) hold for all \( n \in \mathbb{N}_0 \), the existence of \( \beta \in KL \) follows directly from Proposition 9. To show the existence of \( n \in \mathbb{N}_0 \) such that for \( n \geq n \) we either have that \( P_2 = \emptyset \) or \( P_i \neq \emptyset, i \geq 2 \) holds with \( I_p(n, 1) = \emptyset \) for all \( p \in P_i \), suppose that \( P_i \neq \emptyset, i \geq 2 \) holds for some \( n \geq \overline{n} \) and all \( \overline{n} \in \mathbb{N}_0 \) with \( I_p(n, 1) \neq \emptyset \) for some \( p \in P_i \). Then, using Theorem 17 and the existence of \( \beta \in KL \) we obtain a contradiction showing the assertion.

\[ \square \]

Remark 20. While the stability result of Theorem 19 is given for the NMPC case without stabilizing terminal constraints or terminal costs, the only critical component in the proof of this theorem is the condition that \( \bigcup_{P}^{N} P_{\text{ad}}(n, x_p(n), I_p(n)) \neq \emptyset \) which guarantees that the closed loop solution \( x(\cdot) = (x_1(\cdot)^T, \ldots, x_P(\cdot)^T)^T \) exists and satisfies the state constraints. Hence, if instead of the existence conditions of \( \alpha^p_1, \alpha^p_2, \alpha^p_3 \in K_{\infty} \) and \( \alpha > 0 \) such that inequalities (14) and (15) hold we impose other stability conditions – e.g., the terminal constraint condition given in Keerthi and Gilbert [13] or the terminal costs from Chen and Allgöwer [2] – then the same proof can be used to guarantee asymptotic stability of the closed loop.

We like to mention that Algorithm 11 can be extended to an iterative computation of the controls \( u^*_p, p \in P \). To this end only a few steps within the optimization method used to solve the problems of Steps 2b and 2c(iv) are performed. Additionally a second loop containing Steps 2b and 2c is introduced which is terminated if some stopping criterion like the suboptimality based criterion given in Grüne and Pannek [8] is satisfied. Note that the algorithm also allows us to stop agents during such an iterative computation, i.e. if (13) is satisfied for some \( \alpha \geq \overline{\alpha} \in (0, 1) \). Since we allowed for using old and even outdated information in Definition 3, the algorithm even allows to block any computations of some agents for a a certain period depending on the length of an agents prediction without compromising feasibility.

5. Conclusion

We presented a generalized stability result for NMPC controllers without stabilizing terminal constraints or terminal costs. Moreover, we described an algorithm which allows us to run such controllers in a distributed non cooperative setting mostly in parallel. Using only abstract priority and testing maps, we have shown necessary as well as sufficient conditions for stability of the closed loop.
Future research concerning the algorithm will certainly deal with the question how the priority and testing maps should be chosen to minimize the number of priority lists or to maximize the number of controllers that can be run in parallel. From the stability side an indeep analysis is required to apriori guarantee condition (15). The availability of such a condition would then allow us to apriori guarantee Algorithm 11 to asymptotically stabilize the system. One idea in this direction is outlined in Grüne and Worthmann [11, Section 7] and suggests the use of ISS small gain theorems to treat this problem.

6. Acknowledgement

This work was supported by the Leopoldina Fellowship Programme LPDS 2009-36.

References


