Author(s): Zakharov, Alexey & Zattoni, Elena & Yu, Miao & Jämsä-Jounela, Sirkka-Liisa

Title: A performance optimization algorithm for controller reconfiguration in fault tolerant distributed model predictive control

Year: 2015

Version: Post print

Please cite the original version:

Zakharov, Alexey & Zattoni, Elena & Yu, Miao & Jämsä-Jounela, Sirkka-Liisa. 2015. A performance optimization algorithm for controller reconfiguration in fault tolerant distributed model predictive control. Journal of Process Control. Volume 34. 56-69. ISSN 0959-1524 (printed). DOI: 10.1016/j.jprocont.2015.07.006.

A performance optimization algorithm for controller reconfiguration in fault tolerant distributed model predictive control ⋆,⋆⋆

Alexey Zakharov [a], Elena Zattoni [b], Miao Yu [a], Sirkka-Liisa Jämsä-Jounela [a]

[a] *Aalto University School of Chemical Technology, Department of Biotechnology and Chemical Technology, P.O. Box 16100, 00076 Aalto, Finland*

[b] *Department of Electrical, Electronic and Information Engineering "G. Marconi", Alma Mater Studiorum · University of Bologna, 40136 Bologna, Italy*

Abstract

This paper presents a performance optimization algorithm for controller reconfiguration in fault tolerant distributed model predictive control for large-scale systems. After the fault has been detected and diagnosed, several controller reconfigurations are proposed as candidate corrective actions for fault compensation. The solution of a set of constrained optimization problems with different actuator and setpoint reconfigurations is derived by means of an original approach, exploiting the information on the active constraints in the non-faulty subsystems. Thus, the global optimization problem is split into two optimization subproblems, which enables the on-line computational burden to be greatly reduced. Subsequently, the performances of different candidate controller reconfigurations are compared, and the better performing one is selected and then implemented to compensate the fault effects. Efficacy of the proposed approach has been shown by applying it to the benzene alkylation process, which is a benchmark process in distributed model predictive control.

Key words: Distributed model predictive control; fault tolerant control; controller reconfiguration; constrained optimization; alkylation of benzene.

⋆ Corresponding author A. Zakharov

⋆⋆This work is supported by Academy of Finland Project under Grant No. 13138194.

Email addresses: alexey.zakharov@aalto.fi (Alexey Zakharov), elena.zattoni@unibo.it (Elena Zattoni), miao.yu@aalto.fi (Miao Yu), sirkka-liisa.jamsa-jounela@aalto.fi (Sirkka-Liisa Jämsä-Jounela).

1 Introduction

Increased global competition, higher product quality requirements and environmental regulations have forced the process industry to continuously optimize efficiency and profitability. Advanced control strategies, such as model predictive control (MPC), have made it possible to run processes close to the quality and safety limits thereby increasing profitability, whilst ensuring better end product quality and enhancing safety in plants (Qin and Badgwell, 2003). In the engineering practice, one centralized MPC usually cannot handle the whole large-scale process; instead, several MPCs may work together in a distributed manner to exchange the information of each system to achieve the control objectives. To this end, highly efficient distributed control methods have been developed over the past decades. For instance, Scheu and Marquardt (2011) have developed a distributed model predictive control (DMPC) methodology based on a distributed optimization algorithm, which relies on a coordination mechanism using the first-order sensitivities of the objective functions of neighboring systems. This proposed DMPC can effectively reduce the computational burden and overcome possible communication limitations of the centralized MPC. Several other DMPC schemes have been designed based on cooperative game theory (Maestre et al., 2011), bargaining game theory (Alvarado et al., 2011), and serial decomposition of the centralized problem (Negenborn et al., 2008). DMPCs are more and more widely applied to various control systems, such as reactor-separator processes (Liu et al., 2009), alkylation of benzene (Liu et al., 2010), hot-rolled strip laminar cooling processes (Zheng et al., 2009), accelerated cooling process test rig (Zheng et al., 2011), transportation networks (Negenborn et al., 2008), and formation of unicycle robots (Farina et al., 2014). Thus, it has become a common practice to utilize DMPC strategies in large-scale processes (see also Camponogara et al., 2002; Scattolini, 2009; Negenborn and Maestre, 2014).

Conventional control schemes are developed under the assumption that sensors and actuators are free from faults. However, the occurrence of faults causes degradation in the closed-loop performance and also has an impact on safety, productivity and plant economy. As a result, the research focus is shifting towards advanced management of abnormal situations, such as process disturbances and faults, which still provides great possibilities for further improvement of the process efficiency. To this end, fault tolerant control (FTC) has attracted much attention in the area of engineering practice in recent years (see, e.g., Blanke et al., 1997; Mahmoud et al., 2003; Zhang and Jiang, 2008). In this context, fault tolerant model predictive control (FTMPC), which incorporates fault tolerance properties into MPC, has been extensively studied ever since the earlier contribution by Maciejowski (1999). The corrective actions of FTC can be categorized into two types: fault accommodation and controller reconfiguration, whose difference lies in whether the controller settings will

change for the compensation of fault effects or not. In particular, Pranatyasto and Qin (2001) studied the data-based FTC with a simulated fluid catalytic cracking unit, where the sensor faults were detected by principal component analysis and accommodated in the MPC. In Prakash et al. (2002), a fault-accommodation based FTC system was developed on the basis of diagnostic information provided by the generalized likelihood ratio method. In Kettunen et al. (2008), Sourander et al. (2009) and Kettunen and Jämsä-Jounela (2011), various solutions, including data-based FTMPC with fault accommodation, were proposed and tested in a complex dearomatization process.

Despite being an attractive approach, fault accommodation is infeasible in many cases, especially when the ability to control the system degrades because of an actuator fault. As a result, an actuator reconfiguration approach was proposed, aiming to replace the "dropped out" actuator by means of redundant ones. For example, Gani et al. (2007) developed two alternative single-input single-output controls for a polyethylene reactor, manipulating different control variables: the temperature of a feed flow and a catalyst flow rate. In the case of an actuator failure, the control relying on the healthy actuator is applied. Similarly, Chilin et al. (2012a) considered two actuator faults and developed two back-up controls which are applied when the respective fault is discovered. However, in large-scale systems, it is difficult, or even impossible, to develop back-up control strategies for all possible faults, that is why it is an important issue to ensure plant stability under an on-line reconfigured control, while selecting among the candidate reconfigurations. In particular, Gani et al. (2007) determined the stability regions of the alternative controls when an actuator fault occurs, and Chilin et al. (2012a) utilized a modification of MPC to ensure stability. Even though both approaches were able to safeguard stability, a suitable Lyapunov function must be developed in both methods, which makes them difficult to use in case of large-scale systems.

Besides using redundant actuators, another approach to controller reconfiguration consists in defining a new setpoint for the faulty system. Indeed, the nominal process operating conditions can sometimes become infeasible because of the fault and, in such a case, a new operating point must be defined. Thus, Chilin et al. (2012a) proposed to use the feasible steady state closest to the nominal steady state of the system as the new target operating point. Formerly, Gandhi and Mhaskar (2008) had suggested a "safe-parking" approach which selects new operating points from amongst the feasible steady states of the system achievable by the reconfigured control without destabilizing the system. Gandhi and Mhaskar (2009) had also proposed the selection of new operating points of the faulty unit in a way that the downstream units could continue operating at the nominal process conditions and this was implemented as additional constraints imposed on the new operating points of the faulty systems. As a result, the operating point at the moment of fault diagnosis, which is typically close to the nominal steady state, must belong to

the stability region of the reconfigured control that is developed to operate at new process conditions. The drawback is that this makes stability even more difficult to obtain. Therefore, when we have several controller reconfigurations available to compensate for the fault effects, there is a clear demand for more practical solutions to evaluate the possible controller reconfigurations and to select the better performing one in a timely and optimal manner.

Lately, the well-known ability of MPC to achieve offset-free tracking in the presence of plant-model mismatch has been utilized for fault tolerant control development. In Zhang et al. (2013), an improved linear quadratic fault-tolerant control approach has been designed and applied to a batch process with partial actuator faults. A discrete-time augmented model has been considered, with the state including the output tracking error and the change of the state of the actual process model. This approach has been extended to linear systems with an input-output delay in Zhang et al. (2014a,b) and to MPC utilizing the input-output state-space model in Tao et al. (2014). Alternatively, fault tolerance can be achieved through robust control design, which frequently relies on LMIs (Wang et al., 2013a). Moreover, Wang et al. (2013b) has proposed a fault tolerant control approach for batch processes with actuator faults, based on iterative learning control and 2D model representation. The same approach has been formerly applied by Wang et al. (2012) to a batch process with a state delay. Vahid Naghavi et al. (2014) has proposed a decentralized FTMPC, meaning that there is no information exchange between the local controllers relating to subsystems. Both passive and active fault tolerant control designs have been considered. Using Lyapunov function approach, it has been shown that the proposed method guarantees input-to-state stability. In order to facilitate the development of a reconfigured control in case of an actuator fault, Luppi et al. (2015) has focused on the optimization of the control structure, which includes the selection of controlled and manipulated variables as well as their pairings. The fulfillment of a sufficient condition for decentralized integral controllability is searched to guarantee stability. Through the Tennessee Eastman case study, it has been shown that the proposed methodology produces suitable decentralized control structures for reconfigurable FTC systems.

As most of the FTC systems in the literature are based on a centralized MPC for the whole process, there have been only a few attempts to establish a FTC strategy based on DMPC for complex industrial systems (Gandhi and Mhaskar, 2009; Chilin et al., 2010, 2012b). However, in all these works, the distributed control settings are only used in stability analysis, not in the choice of controller reconfigurations. In order to bridge the gap between FTC and DMPC, a framework for the design of a fault tolerant distributed model predictive control (FTDMPC) strategy is presented herein. After the fault has been detected and diagnosed, the key element of the FTDMPC developed in this work is the performance optimization algorithm, which provides the

solution of a set of constrained optimization problems with different, possible actuator and setpoint reconfigurations. The performance optimization algorithm utilizes the information on the active constraints in the non-faulty subsystems and tackles the global MPC optimization problem by splitting it into two nested subproblems. In this way, the on-line computational burden is greatly reduced. Subsequently, the performances of the different candidate controller reconfigurations are compared, the best performing controller is selected and then implemented, so as to compensate the effects of the fault. The effectiveness of the proposed method has been verified by applying it to a benchmark benzene alkylation process (Liu et al., 2010; Scheu and Marquardt, 2011; Chilin et al., 2012a).

On a last introductory note, we underline that our work is focused on actuator faults and we provide the motivations for this. As it can be seen from the literature review presented above, sensor faults are frequently compensated by means of the fault accommodation approach, which relies on a soft sensor or a state estimation with excluding the faulty measurement. Thus, an accommodation-based FTC can be implemented utilizing the well-developed disciplines of data-based soft-sensing and process state estimation. In contrast, fault accommodation is usually unsuitable to handle actuator faults, frequently leading to major control performance degradation. Instead, a partial failure of actuator efficiency can be treated using the passive FTC approach, as is shown, for instance, in Zhang et al. (2013, 2014a). In this case, various robust MPC methods can be employed for FTC development and the Lyapunov function approach is commonly used to ensure stability in the presence of a fault. However, the stuck actuator faults are among the most difficult failures to handle, as only a control reconfiguration is able to compensate fault effects in this case. At the moment, FTCs dealing with actuator faults mostly switch to a pre-developed back-up control after fault detection, and there is little methodology available to support reconfigured control development. Thus, we consider the actuator faults, requiring online control reconfiguration, as a challenging and interesting problem, especially in the case of large-scale processes.

The remainder of this paper is organized as follows. In Section 2, a general idea of FTDMPC is introduced, which focuses on the function of the performance optimization algorithm for the controller reconfiguration. Section 3 is devoted to the DMPC for faultless interconnected systems. Section 4 shows how the formulation of the original DMPC is modified in the presence of a fault, in order to encompass actuator and setpoint reconfiguration, and how the computational burden implied by its solution is reduced with the introduction of suitable, motivated assumptions. In Section 5, simulation results from a benzene alkylation process are provided to demonstrate the effectiveness of the devised approach. Finally, Section 6 outlines the overall conclusions.

Notation: The symbols $\mathbb{R}$, $\mathbb{Z}_0^+$ and $\mathbb{Z}^+$ stand for the sets of real numbers, non-

negative integer numbers and positive integer numbers, respectively. Matrices and linear maps are denoted by capital letters, like A or Ψ. The transpose of A is denoted by $A^\top$. The Moore-Penrose inverse of A is denoted by $A^\dagger$. The symbol $v = \mathrm{vect}\{v_1, v_2, \ldots, v_r\}$ denotes a vector v obtained by concatenating the vectors v_1, v_2, ..., v_r, in order. The symbol $M = \mathrm{diag}\{M_1, M_2, \ldots, M_s\}$ denotes a block-diagonal matrix M, whose blocks on the main diagonal are the matrices M_1, M_2, ..., M_s, in order. The symbols I_n and $O_{m\times n}$ stand for an n-dimensional identity matrix and an $m \times n$ zero matrix, respectively (subscripts are omitted when the dimension can be inferred from the context).

2 Outline of the fault tolerant distributed model predictive control

The fault tolerant distributed model predictive control scheme for large-scale systems devised in this work mainly includes the following elements: distributed model predictive control, fault detection and diagnosis (FDD), controller reconfiguration based on the performance optimization algorithm. The overall structure of FTDMPC is shown in Figure 1. A large-scale system can be divided into different unit processes according to the process topology, which provides a foundation for both FDD methods and DMPC. Based on the breakdown of the overall control objectives into individual objectives within each subsystem, DMPC is designed considering the information exchange between the individual unit processes. At the same time, hierarchical FDD methods are selected based on the intended use of the methods, the systems and their dynamics, and especially the faults and their characteristics, as proposed in our previous work (Jämsä-Jounela et al., 2013).

When the fault is detected and diagnosed, some possible reconfigured controllers can be designed to achieve the control aims in the presence of the faults. For instance, in case of an actuator fault (such as actuator blocking), the faulty actuator is usually replaced by alternative actuators or some actuator constraints are modified, e.g., according to degradation of the faulty actuator capacity. Hence, several reconfigured control settings can be generated for further evaluation.

In case that the current operating condition is infeasible for the faulty plant, a new operating condition needs to be defined according to the control objectives with the fault information provided by the FDD element. The new operating conditions can be selected from the set of steady states of the system under faulty dynamics. As an additional constraint, the target operating conditions in the downstream units must be disturbed as little as possible (Gandhi and Mhaskar, 2009). In particular, one of the requirements is to maintain the set of current active constraints relating to the non-faulty systems as they were at the nominal operating conditions. A group of candidate setpoints can be

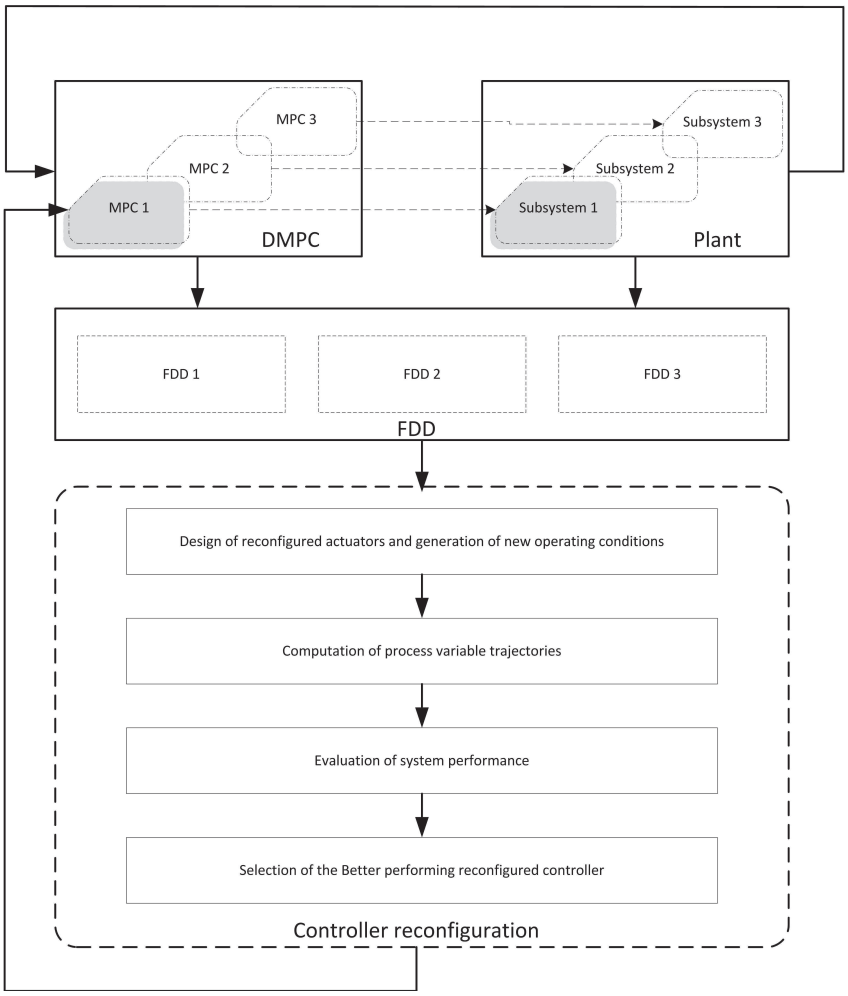

Fig. 1. Outline of fault tolerant distributed model predictive control

found by applying different selection criteria, such as minimizing the distance from the current process state, minimizing the effect on the downstream units, maximizing the economic efficiency of the faulty process unit, minimizing the production rate degradation, etc.

With each possible actuator and setpoint reconfiguration, the MPC turns out to be a different constrained optimization problem. The performance optimization algorithm aims at selecting one of the ensuing corrective actions by evaluating performance of each reconfigured controller action. With the condition that the active constraints in the non-faulty subsystems remain the same as they were in the nominal conditions, the global optimization of MPC can be split into two nested subproblems. As a result, the on-line computational burden is reduced and the consequent selection of the better performing controller can be achieved before the system state is driven far away from the nominal operating conditions. In particular, the criterion for

selecting the controller to be implemented among the various candidates can be based on the definition of some indexes, such as the integral of the error between the predicted trajectories of process variables and their setpoints.

3 The distributed model predictive control problem for the faultless large-scale system

The aim of this section is to introduce the distributed model predictive control problem for the large-scale faultless system. The finite-horizon optimal control problem stated for the discrete-time dynamical system subject to input and output constraints within the prediction horizon is reduced to a constrained algebraic optimization problem, according to techniques extensively used in the literature (see, e.g., Marro et al., 2003; Zattoni, 2008).

The large-scale system consists of the interconnection of a set $\{\Sigma_i,\, i \in \mathcal{I}\}$, with $\mathcal{I} = \{1, 2, \ldots, N\}$, of discrete-time linear time-invariant dynamical systems described by

$$\Sigma_i \equiv \begin{cases} x_i(k+1) = \displaystyle\sum_{j=1}^{N} A_{ij}\, x_j(k) + B_i\, u_i(k), \\ \qquad y_i(k) = C_i\, x_i(k), \end{cases} \qquad i \in \mathcal{I}, \tag{1}$$

where $k \in \mathbb{Z}_0^+$ is the time variable, $x_i \in \mathbb{R}^{n_i}$ is the state, $u_i \in \mathbb{R}^{p_i}$ is the control input, and $y_i \in \mathbb{R}^{q_i}$ is the to-be-controlled output, respectively, with $p_i,\, q_i \leq n_i$ for all $i \in \mathcal{I}$. The matrices A_i, B_i, and C_i are assumed to have constant real entries.

In order to provide an effective formulation of the distributed model predictive control problem for the large-scale system, the following notation is introduced. The symbol $k_c \in \mathbb{Z}^+$ denotes the control horizon and $k_p \in \mathbb{Z}^+$ denotes the prediction horizon. The initial state $x_i(0)$ of Σ_i is denoted by ξ_i, with $i \in \mathcal{I}$, and the symbol $\xi \in \mathbb{R}^n$, with $n = \sum_{i=1}^{N} n_i$, is used to denote the vector of all the initial states: i.e., $\xi = \text{vect}\,\{\xi_1, \xi_2, \ldots, \xi_N\}$. For any Σ_i, with $i \in \mathcal{I}$, the symbols $\boldsymbol{u}_i$ and $\boldsymbol{y}_i$ respectively denote the vectors collecting the sequences of the control inputs over the control horizon and the outputs over the prediction horizon, for the given initial states ξ_i, with $i \in \mathcal{I}$: i.e.,

$$\boldsymbol{u}_i = \text{vect}\,\{u_i(0), u_i(1), \ldots, u_i(k_c - 1)\}, \quad i \in \mathcal{I}, \tag{2}$$

$$\boldsymbol{y}_i = \text{vect}\,\{y_i(1), y_i(2), \ldots, y_i(k_p)\}, \quad i \in \mathcal{I}. \tag{3}$$

Note that, the sequences of the control inputs stop at the time $k = k_p - 1$ (for $k = k_c, \cdots, k_p - 1$, the control inputs remain the same value). And the sequences of the to-be-controlled outputs start at the time $k = 1$. In fact, the

outputs at the initial time $y_i(0)$, with $i \in \mathcal{I}$, are completely determined by the set of the initial states ξ_i, with $i \in \mathcal{I}$. Thus, if the prediction horizon is limited to k_p, the dynamic equations (1), with the initial conditions $x_i(0) = \xi_i$, with $i \in \mathcal{I}$, are equivalent to the algebraic equations

$$\boldsymbol{y_i} = \sum_{j=1}^{N} T_{i,j}\, \boldsymbol{u_j} + V_i\, \xi, \quad i \in \mathcal{I}, \tag{4}$$

where $T_{i,j} \in \mathbb{R}^{k_p\, q_i \times k_c\, p_i}$ and $V_i \in \mathbb{R}^{k_p\, q_i \times n}$ are respectively defined by

$$T_{i,j} = \begin{bmatrix} \tilde{C}_i\, \tilde{B}_j & O & \dots & O \\ \tilde{C}_i\, \tilde{A}\, \tilde{B}_j & \tilde{C}_i\, \tilde{B}_j & \dots & O \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{C}_i\, \tilde{A}^{k_c-1}\, \tilde{B}_j & \tilde{C}_i\, \tilde{A}^{k_c-2}\, \tilde{B}_j & \dots & \tilde{C}_i\, \tilde{B}_j \\ \tilde{C}_i\, \tilde{A}^{k_c}\, \tilde{B}_j & \tilde{C}_i\, \tilde{A}^{k_c-1}\, \tilde{B}_j & \dots & \sum_{l=0}^{1} \tilde{C}_i\, \tilde{A}^{l} \tilde{B}_j \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{C}_i\, \tilde{A}^{k_p-1}\, \tilde{B}_j & \tilde{C}_i\, \tilde{A}^{k_p-2}\, \tilde{B}_j & \dots & \sum_{l=0}^{k_p-k_c} \tilde{C}_i\, \tilde{A}^{l} \tilde{B}_j \end{bmatrix}, \; V_i = \begin{bmatrix} \tilde{C}_i\, \tilde{A} \\ \tilde{C}_i\, \tilde{A}^2 \\ \vdots \\ \tilde{C}_i\, \tilde{A}^{k_p} \end{bmatrix}, \; i, j \in \mathcal{I}, \tag{5}$$

with

$$\tilde{A} = \begin{bmatrix} A_{1,1} & \dots & A_{1,i-1} & A_{1,i} & A_{1,i+1} & \dots & A_{1,N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{i-1,1} & \dots & A_{i-1,i-1} & A_{i-1,i} & A_{i-1,i+1} & \dots & A_{i-1,N} \\ A_{i,1} & \dots & A_{i,i-1} & A_{i,i} & A_{i,i+1} & \dots & A_{i,N} \\ A_{i+1,1} & \dots & A_{i+1,i-1} & A_{i+1,i} & A_{i+1,i+1} & \dots & A_{i+1,N} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{N,1} & \dots & A_{N,i-1} & A_{N,i} & A_{N,i+1} & \dots & A_{N,N} \end{bmatrix}, \; \tilde{B}_i = \begin{bmatrix} O_{n_1 \times p_i} \\ \vdots \\ O_{n_{i-1} \times p_i} \\ B_i \\ O_{n_{i+1} \times p_i} \\ \vdots \\ O_{n_N \times p_i} \end{bmatrix},$$

$$\tilde{C}_i = \begin{bmatrix} O_{q_i \times n_1} & \dots & O_{q_i \times n_{i-1}} & C_i & O_{q_i \times n_{i+1}} & \dots & O_{q_i \times n_N} \end{bmatrix}, \qquad i \in \mathcal{I}.$$

Furthermore, (4) can be written in matrix form as

$$\boldsymbol{y_i} = T_i^*\, \boldsymbol{u}^* + V_i\, \xi, \quad i \in \mathcal{I}, \tag{6}$$

where $T_i^* \in \mathbb{R}^{k_p\, q_i \times k_c\, p}$ and $\boldsymbol{u}^* \in \mathbb{R}^{k_c\, p}$, with $p = \sum_{i=1}^{N} p_i$, are respectively defined by

$$T_i^* = \left[T_{i,1}\ T_{i,2}\ \ldots\ T_{i,N} \right], \quad i \in \mathcal{I}, \tag{7}$$

$$\boldsymbol{u}^* = \text{vect}\,\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_N\}, \tag{8}$$

with $T_{i,j}$ as in (5) and $\boldsymbol{u}_j$ as in (2), for $j \in \mathcal{I}$.

The cost functional is defined by

$$J = \sum_{i=1}^{N} \left\{ \sum_{k=0}^{k_p-1} \left(y_i(k+1) - \eta_i\right)^\top Q_i \left(y_i(k+1) - \eta_i\right) + \sum_{k=0}^{k_c-1} u_i(k)^\top R_i\, u_i(k) \right\}, \tag{9}$$

where $\eta_i \in \mathbb{R}^{q_i}$ is the set point for the output y_i, while $Q_i \in \mathbb{R}^{q_i \times q_i}$ and $R_i \in \mathbb{R}^{p_i \times p_i}$ are positive-definite symmetric matrices, with $i \in \mathcal{I}$. The objective of the optimization, introduced in (9), is indeed the objective of the centralized MPC equivalent to the DMPC. The centralized objective can be split among the local controllers in different ways, but the "natural" splitting should be the most beneficial in practice, as it minimizes the number of necessary DMPC iterations. Since the mathematics behind the splitting method is nontrivial — as can be seen, e.g., from Scheu and Marquardt (2011) — in order to avoid bluring the main idea presented in the next section with too many technical details, we will proceed herein with reference to the objective of the centralize MPC. With the notation introduced in (2) and (3), (9) can be written as

$$J = \sum_{i=1}^{N} \left\{ \left(\boldsymbol{y}_i - \boldsymbol{\eta}_i\right)^\top Q_i^* \left(\boldsymbol{y_i} - \boldsymbol{\eta}_i\right) + \boldsymbol{u_i}^\top R_i^*\, \boldsymbol{u_i} \right\}, \tag{10}$$

where $\boldsymbol{\eta}_i \in \mathbb{R}^{k_p\, q_i}$, $Q_i^* \in \mathbb{R}^{k_p\, q_i \times k_p\, q_i}$, and $R_i^* \in \mathbb{R}^{k_c\, p_i \times k_c\, p_i}$, with $i \in \mathcal{I}$, are respectively given by $\boldsymbol{\eta}_i = \text{vect}\,\{\eta_i, \eta_i, \ldots, \eta_i\}$, $Q_i^* = \text{diag}\,\{Q_i, Q_i, \ldots, Q_i\}$, and $R_i^* = \text{diag}\,\{R_i, R_i, \ldots, R_i\}$. Furthermore, taking (6)–(8) into account, (10) can be written as

$$J = \sum_{i=1}^{N} \left\{ \left(T_i^*\, \boldsymbol{u}^* + V_i\, \xi - \boldsymbol{\eta}_i\right)^\top Q_i^* \left(T_i^*\, \boldsymbol{u}^* + V_i\, \xi - \boldsymbol{\eta}_i\right) + \boldsymbol{u}^{*\top} S_i^*\, \boldsymbol{u}^* \right\}, \tag{11}$$

where $S_i^* \in \mathbb{R}^{k_c\, p \times k_c\, p}$ is given by

$$S_i^* = \text{diag}\,\{\, O_{k_c\, p_1 \times k_c\, p_1}\ \ldots\ O_{k_c\, p_{i-1} \times k_c\, p_{i-1}}\ R_i^*\ O_{k_c\, p_{i+1} \times k_c\, p_{i+1}}\ \ldots\ O_{k_c\, p_N \times k_c\, p_N}\,\}, \; i \in \mathcal{I}.$$

In view of the next developments, it is worth elaborating further on the cost functional, so as to restate it as the sum of a quadratic term in the unknown $\boldsymbol{u}^*$, a linear term in $\boldsymbol{u}^*$, and a constant, as specified below. First, note that,

by means of simple algebraic manipulations, (11) can be written as

$$J = \sum_{i=1}^{N} \left(\boldsymbol{u}^{*\top} \Psi_i \, \boldsymbol{u}^* + \varphi_i^\top \, \boldsymbol{u}^* + \rho_i \right), \tag{12}$$

where $\Psi_i \in \mathbb{R}^{k_c\, p \times k_c\, p}$, $\varphi_i \in \mathbb{R}^{k_c\, p}$, and $\rho_i \in \mathbb{R}$, with $i \in \mathcal{I}$, are respectively defined as follows

$$\Psi_i = T_i^{*\top} \, Q_i^* \, T_i^* + S_i^*, \quad i \in \mathcal{I}, \tag{13}$$
$$\varphi_i = 2\, T_i^{*\top} \, Q_i^* \, (V_i \, \xi - \eta_i), \quad i \in \mathcal{I}, \tag{14}$$
$$\rho_i = \xi^\top \, V_i^\top \, Q_i^* \, V_i \, \xi - 2\, \xi^\top \, V_i^\top \, Q_i^* \eta_i + \eta_i^\top \, Q_i^* \eta_i, \quad i \in \mathcal{I}. \tag{15}$$

Then, by collecting $\boldsymbol{u}^*$ from each term in (12), one gets

$$J = \boldsymbol{u}^{*\top} \, \Psi \, \boldsymbol{u}^* + \varphi^\top \, \boldsymbol{u}^* + \rho, \tag{16}$$

where $\Psi \in \mathbb{R}^{k_c\, p \times k_c\, p}$, $\varphi \in \mathbb{R}^{k_c\, p}$, and $\rho \in \mathbb{R}$ are respectively defined by $\Psi = \sum_{i=1}^{N} \Psi_i$, $\varphi = \sum_{i=1}^{N} \varphi_i$, and $\rho = \sum_{i=1}^{N} \rho_i$.

The control inputs and the to-be-controlled outputs are subject to constraints described by the set of inequalities

$$M_i \, u_i(k) \leq f_i, \quad k = 0, 1, \ldots, k_c - 1, \quad i \in \mathcal{I}, \tag{17}$$
$$N_i \, y_i(k) \leq g_i, \quad k = 1, 2, \ldots, k_p, \quad i \in \mathcal{I}, \tag{18}$$

where $M_i \in \mathbb{R}^{v_i \times p_i}$, $N_i \in \mathbb{R}^{w_i \times q_i}$, $f_i \in \mathbb{R}^{v_i}$, and $g_i \in \mathbb{R}^{w_i}$ are given. Taking (2), (3) into account, one can write (17), (18) in a more compact form as

$$M_i^* \, \boldsymbol{u}_i \leq f_i^*, \quad i \in \mathcal{I}, \tag{19}$$
$$N_i^* \, \boldsymbol{y}_i \leq g_i^*, \quad i \in \mathcal{I}, \tag{20}$$

where $M_i^* \in \mathbb{R}^{k_c\, v_i \times k_c\, p_i}$ and $N_i^* \in \mathbb{R}^{k_p\, w_i \times k_p\, q_i}$ are respectively defined by $M_i^* = \mathrm{diag}\,\{M_i,\, M_i,\, \ldots, M_i\}$ and $N_i^* = \mathrm{diag}\,\{N_i,\, N_i,\, \ldots, N_i\}$, while $f_i^* \in \mathbb{R}^{k_c\, v_i}$ and $g_i^* \in \mathbb{R}^{k_p\, w_i}$ are respectively defined by $f_i^* = \mathrm{vect}\,\{f_i, f_i, \ldots, f_i\}$ and $g_i^* = \mathrm{vect}\,\{g_i, g_i, \ldots, g_i\}$. Moreover, in light of (6)–(8), (19), (20) can be written as

$$K_i^* \, \boldsymbol{u}^* \leq f_i^*, \quad i \in \mathcal{I}, \tag{21}$$
$$N_i^* \, T_i^* \boldsymbol{u}^* + N_i^* \, V_i \, \xi \leq g_i^*, \quad i \in \mathcal{I}, \tag{22}$$

where $K_i^* \in \mathbb{R}^{k_c\, v_i \times k_c\, p}$ is given by

$$K_i^* = \left[O_{k_c\, v_i \times k_c\, p_1} \; \ldots \; O_{k_c\, v_i \times k_c\, p_{i-1}} \; M_i^* \; O_{k_c\, v_i \times k_c\, p_{i+1}} \; \ldots \; O_{k_c\, v_i \times k_c\, p_N} \right], \quad i \in \mathcal{I}. \tag{23}$$

Furthermore, (21), (22) can be grouped into the set of inequalities

$$G_i \, \boldsymbol{u}^* + L_i \, \xi + \ell_i \leq 0, \quad i \in \mathcal{I}, \tag{24}$$

where $G_i \in \mathbb{R}^{(k_c v_i + k_p w_i) \times k_c \, p}$, $L_i \in \mathbb{R}^{(k_c v_i + k_p w_i) \times n}$, and $\ell_i \in \mathbb{R}^{(k_c v_i + k_p w_i)}$ are given by

$$G_i = \begin{bmatrix} K_i^* \\ N_i^* \, T_i^* \end{bmatrix}, \quad L_i = \begin{bmatrix} O_{k_c \, v_i \times n} \\ N_i^* \, V_i \end{bmatrix}, \quad \ell_i = \begin{bmatrix} f_i^* \\ g_i^* \end{bmatrix}, \quad i \in \mathcal{I}. \tag{25}$$

Finally, the set of inequalities (24) can be recast as

$$G \, \boldsymbol{u}^* + L \, \xi + \ell \leq 0, \tag{26}$$

where $G \in \mathbb{R}^{\, z \times k_c \, p}$, $L \in \mathbb{R}^{\, z \times n}$, and $\ell \in \mathbb{R}^{\, z}$, with $z = \sum_{i=1}^{N} (k_c v_i + k_p w_i)$, are given by

$$G = \begin{bmatrix} G_1 \\ G_2 \\ \vdots \\ G_N \end{bmatrix}, \quad L = \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_N \end{bmatrix}, \quad \ell = \begin{bmatrix} \ell_1 \\ \ell_2 \\ \vdots \\ \ell_N \end{bmatrix}. \tag{27}$$

Hence, to summarize, the optimization problem over the prediction time consists in finding $\boldsymbol{u}^*$ so as to minimize the cost functional J, defined by (16), under the constraint (26). The solution to this problem can be sought by exploiting classic results such as Karush-Kuhn-Tucker conditions and by applying the related computational algorithms (see, e.g., Boyd and Vandenberghe, 2004). However, since in model predictive control the optimization is performed within a receding horizon, which implies that the stated problem is to be solved at each time step with the new initial conditions and, in addition, since the systems addressed herein are large-scale systems, which may imply the manipulation of matrices of huge dimensions, ad-hoc algorithms have been developed for distributed model predictive control, like, e.g., that presented in the already mentioned (Scheu and Marquardt, 2011). In particular, Scheu and Marquardt's algorithm is the one that will be employed in Section 5, in combination with the implementation of the idea presented in the next section, thus achieving a dramatic reduction of the computational burden.

4 The distributed model predictive control problem for the faulty large-scale system, with actuator and setpoint reconfiguration

The aim of this section is to show how the approach to the model predictive control problem presented in Section 3 is modified when an actuator fault is

detected in one of the interconnected systems described by (1). In fact, the detection of the fault triggers the so-called reconfiguration process: namely, faulty actuators are replaced by back-up actuators in the faulty system, the setpoints of the to-be-controlled outputs are redetermined, and a new model predictive controller is derived by solving a different optimization problem. Moreover, since the solution of the complete optimization problem, as it turns out by performing the appropriate modifications in the original problem presented in Section 3, may imply a huge computational burden, not sustainable within an on-line reconfiguration process in the presence of a fault, some motivated assumptions are introduced, in order to obtain a simplified version of the optimization problem, suitable to be solved by a real-time operating system.

In particular, as was pointed out by Skogestad (2004), in industrial processes, the optimization is generally subject to constraints and, at the optimum, many of these are usually "active". In this circumstance, if the fault is detected early after the occurrence, the perturbation caused by the fault to the constraints in non-faulty systems is not significant. In other words, the active constraints in these systems remain the same as they were in the nominal operating conditions. In light of these considerations, we will henceforth consider the unknown inputs $\boldsymbol{u}^*$ introduced in Section 3 as displacements with respect to their optimal values in the nominal conditions (this can be made by suitably redefining the origin of the input space) and we will split the original problem into two subproblems.

In brief, the first subproblem consists in the minimization of the cost functional with respect to the sole inputs of the faultless systems, assuming that the active constraints in the faultless systems are known as well as the input to the faulty system. The second subproblem consists in the minimization with respect to the input of the faulty system, taking into account the constraints on the faulty system. Each of these subproblems will be clearly stated in the following developments, as soon as the elements for its formal definitions are made available.

Let us assume that the detected fault has occurred in the system Σ_i, for a known $i \in \mathcal{I}$. The fault tolerant approach developed in this work provides the reconfiguration of the control inputs and the redefinition of the output setpoints in the faulty system. As to the reconfiguration of the control inputs, it is assumed that the control input u_i consists of a set of control inputs which are manipulable when the system is faultless and a set of back-up control inputs which are redundant (hence, not used) in the absence of faults. However, when an actuator fault occurs, some of the manipulable inputs are not available anymore and, therefore, they are replaced by some of the back-up control inputs. In the description of the healthy interconnected systems presented in Section 3, the presence of back-up inputs can be modeled by a suitable defini-

tion of the constraint equation, where, in particular, the matrix G is defined in such a way that the back-up inputs are forced to be equal to zero as long as the interconnected systems are faultless. Viceversa, the reconfiguration of the control inputs can be modelled by redefining the constraint equation in such a way that the faulty inputs are forced to have a constant value (namely, zero, with no loss of generality), while the previous constraint on the back-up inputs is removed. In order to avoid notation clutter, it is assumed henceforth that the constraint equation (26) has been redefined according to the considerations above. As to the redetermination of the output setpoints, this is required whenever the original setpoints cannot be reached anymore, due to the occurrence of the fault, not even with the available redundant actuators. The redetermination of the output setpoints affects the weighting parameters φ and ρ of the cost functional (16). Likewise, it is assumed henceforth that the cost functional (16) has been redefined according to these arguments. Hence, the remainder of this section formalizes the approach to the solution of the optimization problem in a fashion which is suitable for on-line processing.

First, the control inputs collected in the vector $\boldsymbol{u}^*$, defined by (8), are reordered in such a way that the inputs of Σ_i, the faulty system, are placed in the last $k_c\, p_i$ positions, which allows a convenient partition to be introduced in the cost functional and the constraint equations, as is shown in the following. Let the similarity transformation $W \in \mathbb{R}^{k_c\, p \times k_c\, p}$ be defined by

$$W = \begin{bmatrix} I_{r_1} & O & O \\ O & O & I_{r_2} \\ O & I_{r_3} & O \end{bmatrix}, \tag{28}$$

with $r_1 = k_c \sum_{j=1}^{i-1} p_j$, $r_2 = k_c \sum_{j=i+1}^{N} p_j$, and $r_3 = k_c\, p_i$. It is worth noting that $W = W^{-1} = W^\top$. Let $\boldsymbol{u}^{*\prime}$ denote the input vector with respect to the new coordinates, so that

$$\boldsymbol{u}^* = W\, \boldsymbol{u}^{*\prime}. \tag{29}$$

Then, in light of (29), the cost functional (16) can be written as

$$J = \boldsymbol{u}^{*\prime\top}\, \Psi'\, \boldsymbol{u}^{*\prime} + \varphi'^{\top}\, \boldsymbol{u}^{*\prime} + \rho, \tag{30}$$

where $\Psi' = W\, \Psi\, W$ and $\varphi' = W\, \varphi$. Thus, if Ψ and φ, partitioned according to (28), are given by

$$\Psi = \begin{bmatrix} \Psi_{11} & \Psi_{12} & \Psi_{13} \\ \Psi_{12}^\top & \Psi_{22} & \Psi_{23} \\ \Psi_{13}^\top & \Psi_{23}^\top & \Psi_{33} \end{bmatrix}, \quad \varphi = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{bmatrix}, \tag{31}$$

with respect to new coordinates, Ψ' and φ' are given by

$$\Psi' = \left[\begin{array}{cc|c} \Psi_{11} & \Psi_{13} & \Psi_{12} \\ \Psi_{13}^\top & \Psi_{33} & \Psi_{23}^\top \\ \hline \Psi_{12}^\top & \Psi_{23} & \Psi_{22} \end{array}\right], \quad \varphi' = \left[\begin{array}{c} \varphi_1 \\ \varphi_3 \\ \hline \varphi_2 \end{array}\right]. \tag{32}$$

With respect to the new coordinates, the last $k_c\, p_i$ components of the input $\boldsymbol{u}^{*\prime}$ concern the faulty system Σ_i, while the former components concern the faultless systems Σ_j, with $j \in \mathcal{I}$, $j \neq i$. According to this, let

$$\boldsymbol{u}^{*\prime} = \left[\begin{array}{c} \boldsymbol{u}_h^* \\ \boldsymbol{u}_f^* \end{array}\right]. \tag{33}$$

Accordingly, Ψ' and φ' in (32) can be written in more compact form as

$$\Psi' = \left[\begin{array}{cc} \Psi_{hh} & \Psi_{hf} \\ \Psi_{hf}^\top & \Psi_{ff} \end{array}\right], \quad \varphi' = \left[\begin{array}{c} \varphi_h \\ \varphi_f \end{array}\right]. \tag{34}$$

With the notation introduced in (33) and (34), the cost functional (30) can be written as

$$J = \boldsymbol{u}_h^{*\top}\, \Psi_{hh}\, \boldsymbol{u}_h^* + 2\, \boldsymbol{u}_f^{*\top}\, \Psi_{hf}^\top\, \boldsymbol{u}_h^* + \boldsymbol{u}_f^{*\top}\, \Psi_{ff}\, \boldsymbol{u}_f^* + \varphi_h^\top\, \boldsymbol{u}_h^* + \varphi_f^\top\, \boldsymbol{u}_f^* + \rho. \tag{35}$$

A similar reasoning can be applied to the constraint (26). In fact, taking (29) into account, one can write (26) as

$$G'\, \boldsymbol{u}^* + L\, \xi + \ell \leq 0, \tag{36}$$

where $G' = G\, W$ and, according to the partition (33), (36) can be written as

$$G_h\, \boldsymbol{u}_h^* + G_f\, \boldsymbol{u}_f^* + L\, \xi + \ell \leq 0. \tag{37}$$

Then, at first, the assumption of taking into account only active constraints in non-faulty systems is introduced, which means that (37) is replaced by

$$F_h\, \boldsymbol{u}_h^* + F_f\, \boldsymbol{u}_f^* + E\, \xi + d = 0, \tag{38}$$

where F_h, F_f, E, and d have been respectively extracted from G_h, G_f, L, and ℓ by only considering equality constraints in the faultless systems. Moreover, the cost functional (35) is minimized with respect to the control inputs $\boldsymbol{u}_h^*$ of the sole faultless systems.

Namely, *the first optimization problem* which is tackled is stated as follows.

Problem 1 *Find $\boldsymbol{u}_h^*$ such that J, given by (35), is minimized, under the constraint (38).*

The Lagrangian function for the problem stated above is defined by

$$\mathcal{L}(\boldsymbol{u}_h^*, \lambda) = \lambda^\top (F_h\, \boldsymbol{u}_h^* + F_f\, \boldsymbol{u}_f^* + E\, \xi + d) + \\ \boldsymbol{u}_h^{*\top}\, \Psi_{hh}\, \boldsymbol{u}_h^* + 2\, \boldsymbol{u}_f^{*\top}\, \Psi_{hf}^\top\, \boldsymbol{u}_h^* + \boldsymbol{u}_f^{*\top}\, \Psi_{ff}\, \boldsymbol{u}_f^* + \varphi_h^\top\, \boldsymbol{u}_h^* + \varphi_f^\top\, \boldsymbol{u}_f^* + \rho,$$

where λ denotes the vector of the Lagrange multipliers. Then, according to the Lagrangian multiplier approach, the solution of the following system of equations is sought:

$$\begin{cases} 2\, \boldsymbol{u}_h^{*\top}\, \Psi_{hh} + 2\, \boldsymbol{u}_f^{*\top}\, \Psi_{hf}^\top + \varphi_h^\top + \lambda^\top\, F_h = 0, \\ F_h\, \boldsymbol{u}_h^* + F_f\, \boldsymbol{u}_f^* + E\, \xi + d = 0. \end{cases} \tag{39}$$

Since Ψ_{hh} is symmetric positive-definite, the unknown $\boldsymbol{u}_h^*$ can be made explicit from the first of (39) and replaced in the second. Thus, (39) provide

$$\begin{cases} \boldsymbol{u}_h^* = -\Psi_{hh}^{-1}\, \Psi_{hf}\, \boldsymbol{u}_f^* - \dfrac{1}{2}\, \Psi_{hh}^{-1}\, \varphi_h - \dfrac{1}{2}\, \Psi_{hh}^{-1}\, F_h^\top\, \lambda, \\ \lambda = 2\, \Theta\, (F_f - F_h\, \Psi_{hh}^{-1})\, \Psi_{hf})\, \boldsymbol{u}_f^* - \Theta\, F_h \Psi_{hh}^{-1}\, \varphi_h + 2\, \Theta\, E\, \xi + 2\, \Theta\, d, \end{cases} \tag{40}$$

where $\Theta = (F_h\, \Psi_{hh}\, F_h^\top)^\dagger$. Then, by replacing the second of (40) in the first, one gets the optimal value for $\boldsymbol{u}_h^*$ as

$$\boldsymbol{u}_h^* = \Gamma\, \boldsymbol{u}_f^* + \gamma, \tag{41}$$

where

$$\Gamma = -\Psi_{hh}^{-1}\, \Big(\Psi_{hf} + F_h^\top\, \Theta\, (F_f - F_h\, \Psi_{hh}^{-1}\, \Psi_{hf})\Big), \tag{42}$$

$$\gamma = -\Psi_{hh}^{-1}\, \left(-\frac{1}{2}(I + F_h^\top\, \Theta\, F_h\, \Psi_{hh}^{-1})\, \varphi_h - F_h^\top\, \Theta\, (E\, \xi + d)\right). \tag{43}$$

Furthermore, by replacing (41) in (35), one gets

$$J = \boldsymbol{u}_f^{*\top}\, \Phi\, \boldsymbol{u}_f^* + 2\, \sigma^\top\, \boldsymbol{u}_f^* + \kappa, \tag{44}$$

where

$$\Phi = \Gamma^\top\, \Psi_{hh}\, \Gamma + \Psi_{hf}^\top\, \Gamma + \Psi_{ff}, \tag{45}$$

$$\sigma = \Gamma^\top\, \Psi_{hh}\, \gamma + \Psi_{hf}^\top\, \gamma + \Gamma^\top \varphi_h, \tag{46}$$

$$\kappa = \gamma^\top\, \Psi_{hh}\, \gamma + \varphi_h^\top\, \gamma + \rho. \tag{47}$$

Moreover, by replacing (41) in (37), one gets

$$\Lambda \, \boldsymbol{u}_f^* + L \, \xi + \mu \leq 0, \tag{48}$$

where

$$\Lambda = G_h \, \Gamma + G_f, \tag{49}$$
$$\mu = G_h \, \gamma + \ell. \tag{50}$$

Hence, *the second optimization problem* is stated as follows.

Problem 2 *Find $\boldsymbol{u}_f^*$, such that J, given by (44), is minimized, under the constraint (48).*

Although the solution of this optimization problem can be obtained by applying Karush-Kuhn-Tucker conditions and the related algorithms, as the original problem presented in Section 3, here the unknown variable $\boldsymbol{u}_f^*$ consists of a subvector of the unknown $\boldsymbol{u}^*$ of the original problem. So, a substantial reduction of the computational complexity has been obtained by means of the devised approach.

In order to better highlight the impact of the reduction of the computational burden achieved by the proposed approach, it is worthwhile stressing that the optimization problem considered above has to be solved for different choices of the actuator and setpoint reconfiguration, so that a set of candidate reconfigured controllers are obtained. Moreover, as is required in MPC, this algorithm has to be iterated at each step of the prediction horizon.

As to the selection of the better performing reconfigured controller, this can be straightforwardly done by comparing the optimal values of the performance indexes, like the Integral Absolute Error (IAE), obtained for each of the candidate reconfigured controllers.

5 Simulation results

This section illustrates the main results obtained by testing the FTDMPC scheme devised in this work on the benzene alkylation process. First, the benchmark process is described briefly, then, the DMPC and FDD methods are introduced separately. Finally, the emphasis is given to FTC, especially with regard to the implementation of the performance optimization algorithm.

5.1 Process description

The process of alkylation of benzene with ethylene to produce ethylbenzene is widely used in the petrochemical industry (see, e.g., Liu et al., 2010). Dehydration of the product produces styrene, which is the precursor to polystyrene and many copolymers. We consider the simulated chemical process for the alkylation of benzene from Scheu and Marquardt (2011) — also depicted in Fig. 2 — to illustrate the performance of the proposed FTDMPC scheme. The plant consists of five units: i.e., four continuous stirred-tank reactors (CSTRs) and one flash separator. The CSTR 1, CSTR 2, and CSTR 3 are in series and involve the alkylation of benzene with ethylene. Pure benzene is fed from stream F_1 and pure ethylene is fed from streams F_2, F_4, and F_6. Two catalytic reactions take place in CSTR 1, CSTR 2, and CSTR 3. Benzene (A) reacts with ethylene (B) and produces the required product ethylbenzene (C) (reaction 1); ethylbenzene can further react with ethylene to form 1,3-diethylbenzene (D) (reaction 2) which is the byproduct. The effluent of CSTR 3, including the products and leftover reactants, is fed to a flash tank separator, in which most of benzene is separated overhead by vaporization and condensation techniques and recycled back to the plant and the bottom product stream is removed. A portion of the recycle stream F_{r2} is fed back to CSTR 1 and another portion of the recycle stream F_{r1} is fed to CSTR 4 together with an additional feed stream F_{10} which contains 1,3-diethylbenzene from further distillation process that we do not consider in this example. In CSTR 4, reaction 2 and catalyzed transalkylation reaction in which 1,3-diethylbenzene reacts with benzene to produce ethylbenzene (reaction 3) takes place. All chemicals left from CSTR 4 eventually pass into the separator. All the materials in the reactions are in liquid phase due to high pressure.

The mathematical model consists of material balances for each component and an energy balance for each unit of the plant, which results in a system model that includes a total of 25 states. The states of the process consist of the concentrations of A, B, C and D in each of the five units and the temperatures of the units. In addition, the model includes nonlinear reaction kinetics as well as a nonlinear description of the phase equilibrium in the flash separator, leading to a total of approximately 100 equations. The state is assumed to be available continuously to the controllers. Ideal liquid and gas phases are assumed to be in equilibrium. All CSTRs are assumed to be well mixed. The pressure in the reactors is assumed to be constant. Each of the units has an external heat/coolant input. In the normal condition, the manipulated inputs to the process are the heat injected to or removed from the five units, Q_1, Q_2, Q_3, Q_4 and Q_5 (u_1, u_2, u_3, u_4 and u_5, respectively). The feed stream flow rates to CSTR 2 and CSTR 3, F_4 and F_6, are the back-up manipulated variables (u_6 and u_7) which are activated for the controller reconfiguration when a fault is detected. The steady-state inputs, u_{is}, $i = 1, \cdots, 7$, as well as the steady-state

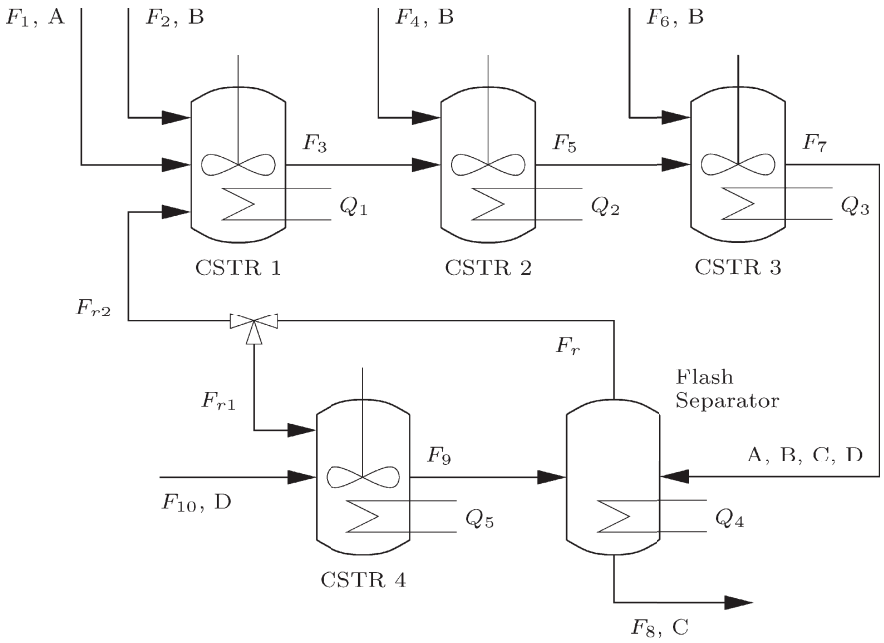

Fig. 2. Process flow diagram for alkylation of benzene (Scheu and Marquardt, 2011)

Table 1
Steady-State Inputs and Temperatures

$u_{1s} = -2.0 \times 10^6\,\mathrm{J/s}$	$u_{7s} = 8.697 \times 10^{-4}\,\mathrm{m^3/s}$
$u_{2s} = -2.0 \times 10^6\,\mathrm{J/s}$	$T_{1s} = 472.32\,\mathrm{K}$
$u_{3s} = -2.0 \times 10^6\,\mathrm{J/s}$	$T_{2s} = 472.35\,\mathrm{K}$
$u_{4s} = 4.1 \times 10^6\,\mathrm{J/s}$	$T_{3s} = 472.39\,\mathrm{K}$
$u_{5s} = -0.01 \times 10^6\,\mathrm{J/s}$	$T_{4s} = 472.00\,\mathrm{K}$
$u_{6s} = 8.697 \times 10^{-4}\,\mathrm{m^3/s}$	$T_{5s} = 472.49\,\mathrm{K}$

temperatures in the five units are shown in Table 1.

The nonlinear model is linearized (by a finite-difference approach) at this operating point, so that the following linear time-invariant model is obtained:

$$\Delta \dot{x} = A\,\Delta x + B\,\Delta u, \quad \Delta x(0) = x_0 - x_s, \tag{51}$$

where $\Delta x = x - x_s$ and $\Delta u = u - u_s$ indicate the deviations of the state and input variables from the steady values (x_s, u_s), and x_0 indicates the initial condition of the plant. The linearized model is used as the internal model of the controller, while the nonlinear model is used to simulate the plant.

Table 2
Constraints on Manipulated Inputs and Temperatures

$\lvert u_1\rvert \leq 0.75\,\mathrm{MJ/s}$	$\lvert u_7\rvert \leq 2 \times 10^{-3}\,\mathrm{m^3/s}$
$\lvert u_2\rvert \leq 0.5\,\mathrm{MJ/s}$	$471\,\mathrm{K} \leq T_1 \leq 474\,\mathrm{K}$
$\lvert u_3\rvert \leq 0.5\,\mathrm{MJ/s}$	$471\,\mathrm{K} \leq T_2 \leq 474\,\mathrm{K}$
$\lvert u_4\rvert \leq 0.6\,\mathrm{MJ/s}$	$471\,\mathrm{K} \leq T_3 \leq 474\,\mathrm{K}$
$\lvert u_5\rvert \leq 0.6\,\mathrm{MJ/s}$	$471\,\mathrm{K} \leq T_4 \leq 474\,\mathrm{K}$
$\lvert u_6\rvert \leq 2 \times 10^{-3}\,\mathrm{m^3/s}$	$471\,\mathrm{K} \leq T_5 \leq 474\,\mathrm{K}$

5.2 Distributed MPC strategy

In this work, the sensitivity-driven DMPC in (Scheu and Marquardt, 2011) is used as the base controller for the alkylation of benzene process. The whole system is divided into two groups, one includes CSTR 1, CSTR 2 and CSTR 3, the other contains CSTR 4 and the flash separator. Thus, the process is under the control of two distributed controllers, and information is exchanged between them. In the non-faulty situation, only inputs u_1, u_2, u_3, u_4 and u_5 are actuated, which means the first distributed controller (DMPC 1) controls the values of Q_1, Q_2 and Q_3, while the second distributed controller (DMPC 2) controls the values of Q_4 and Q_5. When the inputs u_6 and u_7 are actuated in the faulty situation, they are used to replace the corresponding faulty actuators.

For each unit i of the plant, the following conventional objective function is considered:

$$\Phi_i = \frac{1}{2}\int_{t_0}^{t_f} \left(\Delta\, y_i^\top\, Q_i\, \Delta\, y_i + \Delta\, u_i^\top\, R_i\, \Delta\, u_i\right)\, dt, \tag{52}$$

where Q_i and R_i are positive-definite weighting matrices. The inputs $\Delta\, u_i(t)$ are discretized as piecewise-constant functions with sampling time $t = 10\,\mathrm{s}$. The control horizon is assumed to consist of 5 steps, which is enough to achieve good DMPC performance. Furthermore, a prediction horizon of 20 steps, sufficient to ensure the DMPC stability, is used in this work.

The constraints on the manipulated inputs and the temperatures are shown in Table 2. It is worth mentioning that, in addition to the input constraints, the temperatures in the five units were also bounded, in order to keep the process conditions close to the nominal point. The sensitivity-driven DMPC has been tested with setpoint changes. The original setpoint for temperatures was set as in Table 1, then at $t = 100\,\mathrm{s}$ (sample 10), the setpoint was changed to $T_{4s} = 471\,\mathrm{K}$ and $T_{5s} = 474\,\mathrm{K}$, which are close to the boundary, while T_{1s}, T_{2s}

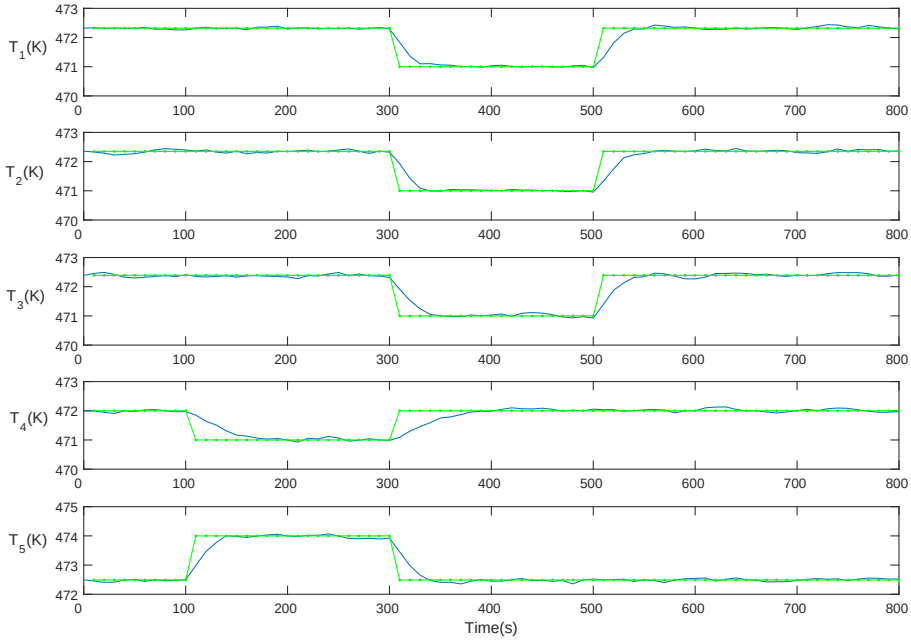

Fig. 3. Test of DMPC with setpoint changing (green dash-dot line: setpoint; blue solid line: temperature variations)

and T_{3s} remain the same as in Table 1. At $t = 300\,\mathrm{s}$ (sample 30), a new setpoint was defined as $T_{1s} = T_{2s} = T_{3s} = 471\,\mathrm{K}$, while T_{4s} and T_{5s} remain the same as in Table 1. Finally, at $t = 500\,\mathrm{s}$ (sample 50), the setpoint was changed back to the original one, as in Table 1. The test result is shown in Figure 3 and it clearly demonstrates that the transition to the new setpoint takes about 100 s for every setpoint change, after which accurate tracking is achieved.

5.3 Fault detection and diagnosis approach

In this work, we utilize the approach of Chilin et al. (2012a) for fault detection and diagnosis in the actuators. A filter is designed for each state and for the k-th, $k = 1, \ldots, 25$, state in the system state x, the filter is designed as follows:

$$\dot{\hat{x}}_k = \bar{A}_k \hat{X}_k + \bar{B}_k \bar{u}_k(\hat{X}_k), \tag{53}$$

where $\hat{x}_k$ is the filter output for the k-th state, $\bar{A}_k$ is the k-th row of $\tilde{A}$, $\bar{B}_k$ is the k-th row of $B = \mathrm{diag}\{B_1, B_2\}$. The input $\bar{u}_k(\hat{X}_k) = \mathrm{vect}\left\{u_1(\hat{X}_k), u_2(\hat{X}_k)\right\} \in \Re^p$ are the distributed controllers based on the sensitivity-driven distributed optimization in the previous subsection, while the actual system states x is replaced with filter states $\hat{X}_k \in \Re^n$. The

state $\hat{X}_k$ is obtained from both the actual state measurements x and the filter output $\hat{x}_k$, as follows:

$$\hat{X}_k(t) = [\, x_1(t), \ldots, x_{k-1}(t), \hat{x}_k(t), x_{k+1}(t), \ldots, x_{25}(t) \,]^\top . \tag{54}$$

The states of the fault detection filters are initialized at $t = 0$ to the actual state values, $\hat{x}_k(0) = x_k(0)$. The information generated by the filters provides a fault-free estimate of the real state at any time t and allows detection of the faults. For each state associated with a filter, the fault detection residual can be defined as

$$r_k(t) = |\hat{x}_k(t) - x_k(t)| , \tag{55}$$

where $k = 1, \ldots, 25$. The residual r_k is easy to obtain because $\hat{x}_k$ is known for all t and the state measurement x_k is also available for all t. If no faults occur, the filter states track the system states, so that $r_k(t) = 0$ for all times. When there is a fault in the system, filter residuals directly affected by the fault will deviate from zero soon after the occurrence of the fault.

In order to avoid false alarms due to the process and sensor measurement noise, thresholds are necessary in the filters. In order to select the detection threshold, the residual distribution has been estimated by simulating the process with noise at the nominal steady state for 1000 steps. The threshold value was determined to ensure that the probability of false detection is nearly zero.

Concerning the synthesis of the residual generators, it is worth noting that, as this work considers only actuator faults, five residuals are enough for fault detection and isolation. In particular, the effect of actuator faults on the temperature in the reactors is very clear, and that is why the corresponding states were used to create the filters. On the other hand, the similar filters for the rest of the states are not required for fault isolation. Since the inputs u_1, u_2, u_3, u_4 and u_5 correspond to the temperatures T_1, T_2, T_3, T_4 and T_5 directly, the following thresholds in the five state filters are set:

$$r_i(t) = \left| \hat{T}_i(t) - T_i(t) \right| < 1\,\mathrm{K}, \quad i = 1, \cdots, 5. \tag{56}$$

When the difference between the state estimate and the measured state exceeds 1 K, the actuator corresponding to the unusual temperature value can be easily identified as faulty. After that, the fault parameter estimation approach outlined in (Chilin et al., 2012a) can be applied to estimate the magnitude of the fault.

5.4 Testing of the performance optimization-based FTDMPC

In this part, two case studies are provided: the first one is to evaluate the candidate reconfigured actuators, while the second one is to check the newly defined operating point.

Finally, some considerations on the benefits of using the devised algorithm in terms of reduction of the computational complexity — hence, in terms of decrease of the CPU time — have been presented.

5.4.1 Case study 1: Evaluating candidate actuator reconfigurations

Firstly, the current operating point is checked to determine if it is feasible under the original control strategy when a fault is diagnosed. Subsequent to this, the performance of the candidate actuator reconfigurations is evaluated.

We consider an actuator fault occurring at $t = 300\,\mathrm{s}$ (sample 30): u_2 is blocked at 95 % of its steady-state value, that is, $u_2 = -1.9 \times 10^6\,\mathrm{J/s}$. Obviously, the temperature in CSTR 2 will be increasing from that time if no FTC is implemented. Figure 4 shows the residual value of the fault detection filter for T_2. At time $t = 310\,\mathrm{s}$ (sample 31), the residual for T_2 exceeds the threshold 1 K, therefore it can be concluded that there is an actuator fault in u_2. At first, we want to check whether the current operating point is feasible under the existing control configuration. That means, DMPC 1 controls the actuators u_1 and u_3, DMPC 2 controls the actuators u_4 and u_5, while u_2 is blocked at $-1.9 \times 10^6\,\mathrm{J/s}$, u_6 and u_7 stay the same as steady-state values.

Figure 5 shows the test result with existing actuators and current operating point. At time $t = 310\,\mathrm{s}$ (sample 31), the performance optimization algorithm is implemented to give the predictions of temperature trajectory for the future 20 steps. It shows directly that the current operating point is not feasible without changing the controller configuration, which is verified by the result under DMPC.

Since the current operating point is not feasible with the existing actuators, one possible solution is to activate another actuator in order to compensate for the efficiency loss in u_2. To demonstrate the function of the performance optimization algorithm for controller reconfiguration, two back-up actuator reconfigurations are investigated. The first is to activate the feed stream flow rates to CSTR 2, u_6, and the second is to activate the feed stream flow rates to CSTR 3, u_7. In the first case, DMPC 1 controls the actuators u_1, u_3 and u_6, DMPC 2 controls the actuators u_4 and u_5, while u_2 is blocked at $-1.9 \times 10^6\,\mathrm{J/s}$, u_7 stays the same as steady-state values. In the second case, DMPC 1 controls the actuators u_1, u_3 and u_7, DMPC 2 controls the actuators

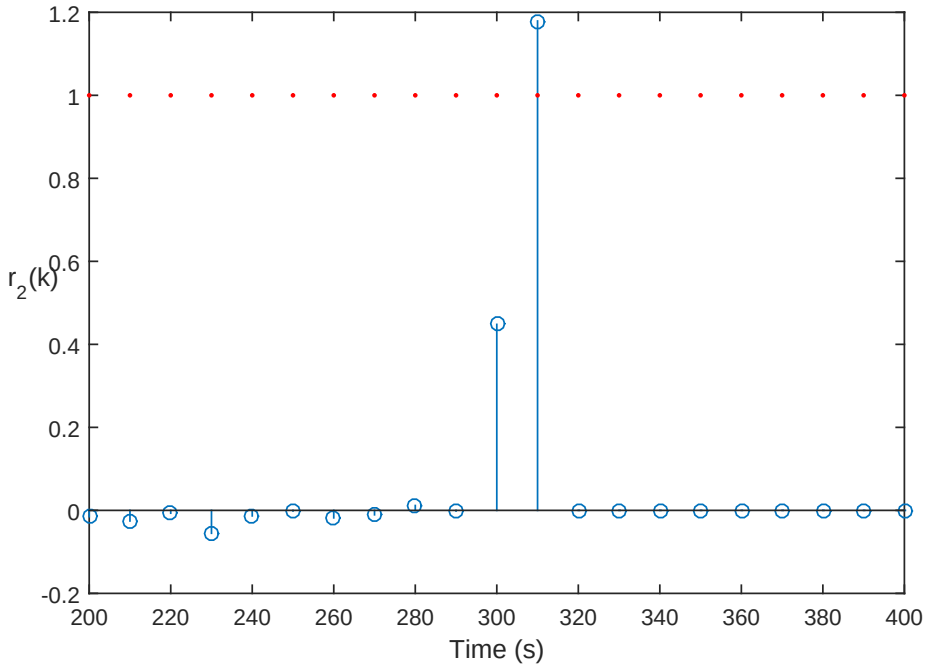

Fig. 4. Fault detection filter residuals for T_2

u_4 and u_5, while u_2 is blocked at -1.9×10^6 J/s, u_6 stays the same as steady-state values.

Figure 6 and Figure 7 depict the test result with the activation of u_6 and u_7 under the current operating point, respectively. From the trajectories under performance optimization algorithm shown in Figure 6, it is easy to see that the temperatures can be driven to setpoint after 12 steps under the effect of u_6. While the trajectories under performance optimization algorithm shown in Figure 7 demonstrate irrefutably that activating u_7 does not make much difference compared with Figure 5. After the comparison, it can be decided to implement the first control reconfiguration at $t = 310$ s (sample 31), which will result in the temperatures converging to the setpoint with reconfigured controller at $t = 430$ s (sample 43).

5.4.2 Case study 2: Checking newly defined operating point

In this part, a case study where the current operating point is not feasible with either original control strategy or any reconfigured actuators is outlined. Hence, another operating point must be designed, based on the characteristics of the fault. We consider an actuator fault occurring at $t = 300$ s (sample 30): u_1 is blocked at 97.5 % of its steady-state value, that is, $u_1 = -1.95 \times 10^6$ J/s and obviously, the temperature in CSTR 1 will be increasing from that time.

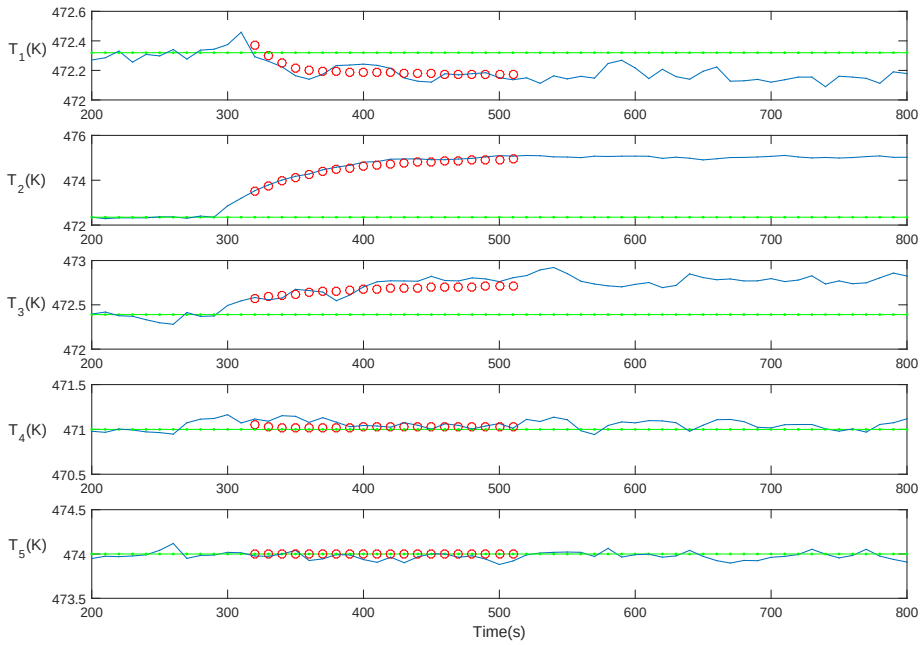

Fig. 5. Test results with existing actuators and current operating point (green dash-dot line: setpoint; blue solid line: temperature variations under DMPC; red circle line: trajectories under performance optimization algorithm)

Figure 8 shows the residual value of fault detection filter for T_1. At time $t = 320\,\mathrm{s}$ (sample 32), the residual exceeds the threshold 1 K, thus, it can concluded that there is an actuator fault in u_1. It is clear that the fault in u_1 in CSTR 1 cannot be compensated by current control strategy or activating u_6 and u_7 in CSTR 2 and 3. The trajectories under performance optimization algorithm for the future 20 steps at time $t = 320\,\mathrm{s}$ (sample 32) has verified our supposition as in Figure 9. Three different control configurations were utilized to carry out the test:

- the first controller uses the current control configuration, that is, DMPC 1 controls the actuators u_2 and u_3, DMPC 2 controls the actuators u_4 and u_5, while u_1 is blocked at $-1.95 \times 10^6\,\mathrm{J/s}$, u_6 and u_7 stay the same as the steady-state values;
- the second controller activates the feed stream flow rates to CSTR 2, u_6, that is, DMPC 1 controls the actuators u_2, u_3 and u_6, DMPC 2 controls the actuators u_4 and u_5, while u_1 is blocked at $-1.95 \times 10^6\,\mathrm{J/s}$, u_7 stays the same as the steady-state values;
- the third controller activates the feed stream flow rates to CSTR 3, u_7, that is, DMPC 1 controls the actuators u_2, u_3 and u_7, DMPC 2 controls the actuators u_4 and u_5, while u_1 is blocked at $-1.95 \times 10^6\,\mathrm{J/s}$, u_6 stays the same as the steady-state values.

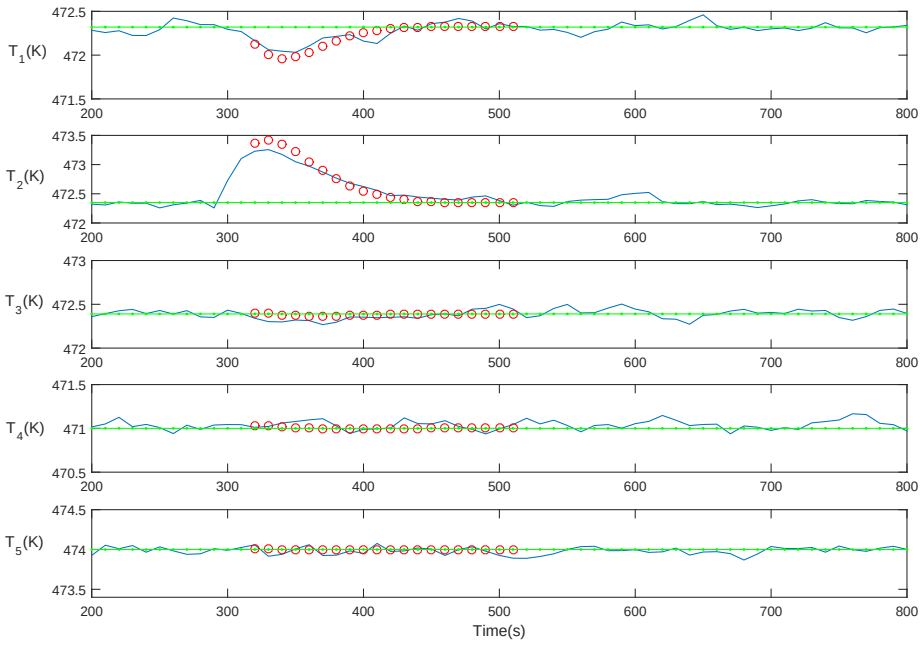

Fig. 6. Test results obtained by activating u_6 under the current operating point (green dash-dot line: setpoint; blue solid line: temperature variations under DMPC; red circle line: trajectories under performance optimization algorithm)

As can be seen from Figure 9, all the three control configurations cannot drive the temperature in CSTR 1, T_1, to the current setpoint as it inevitably increases with the loss of efficiency in u_1. Thus, one possible solution is to increase the setpoint for T_1 within the constraints detailed in Table 2. Another choice is to decrease the setpoint for the temperature in the flash separator, T_4. Since the recycled vapor stream goes from flash separator to CSTR 1, the cooling of this stream can also lead to the decreasing of T_1. The new operating point is designed as follows: $T_{1s} = 473.36\,\mathrm{K}$, $T_{2s} = 472.35\,\mathrm{K}$, $T_{3s} = 472.39\,\mathrm{K}$, $T_{4s} = 471.00\,\mathrm{K}$, $T_{5s} = 473.00\,\mathrm{K}$.

Figure 10 shows the trajectories under performance optimization algorithm for the future 20 steps with the newly designed setpoint, at time $t = 320\,\mathrm{s}$ (sample 32). It can be easily seen that both the second and the third controllers can obtain very good performance. After checking the difference between the predicted trajectory and the setpoint, it was found that the third controller performs slightly better than the second one and, as a result, u_7 is activated at time $t = 320\,\mathrm{s}$ (sample 32). The test result with the activation of u_7 and the newly designed operating point is shown in Figure 11. The temperature trajectory tracks the newly designed setpoint very well and the trajectory under performance optimization algorithm is close to the actual temperature.

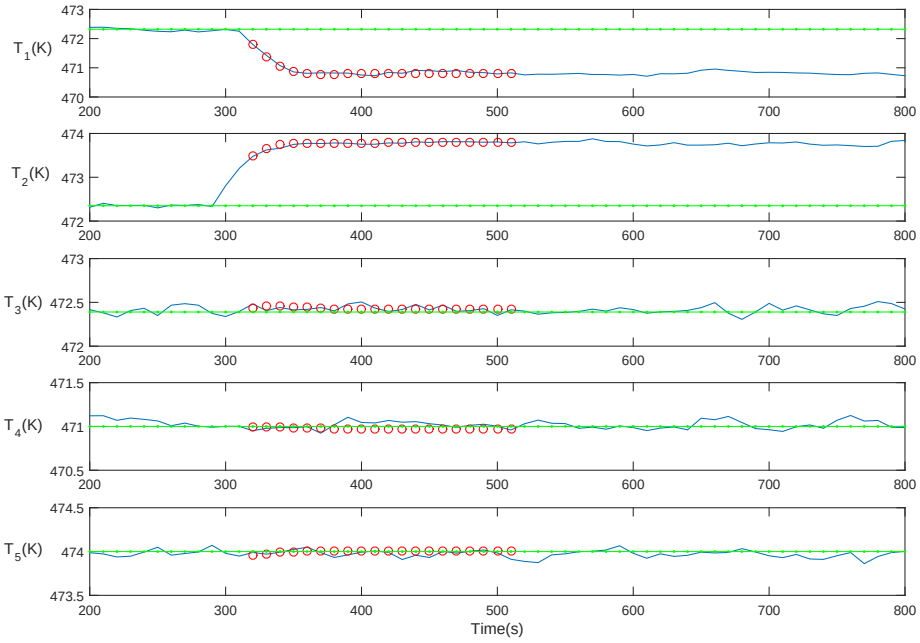

Fig. 7. Test results obtained by activating u_7 under the current operating point (green dash-dot line: setpoint; blue solid line: temperature variations under DMPC; red circle line: trajectories under performance optimization algorithm)

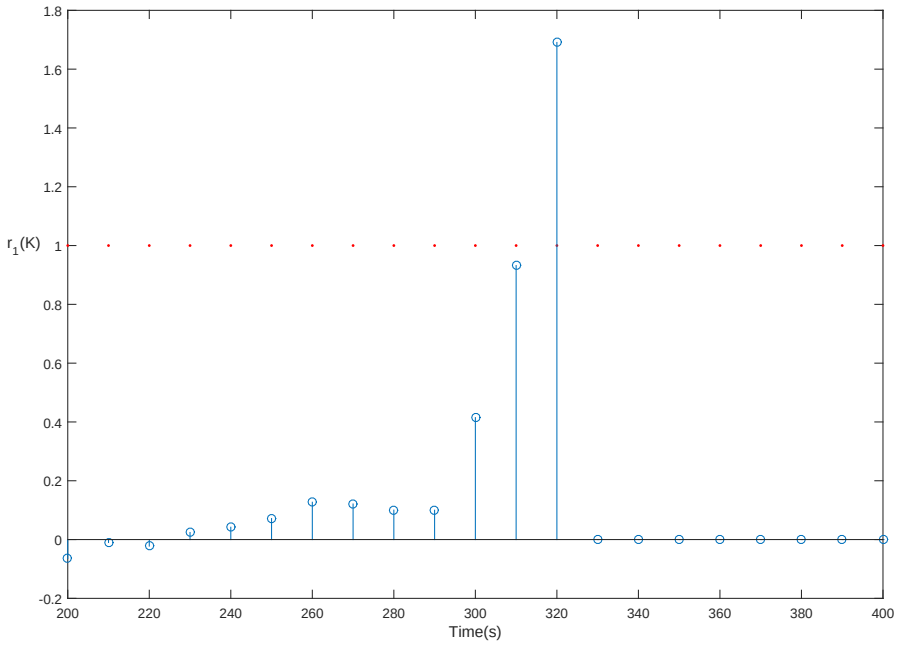

Fig. 8. Fault detection filter residuals for T_1

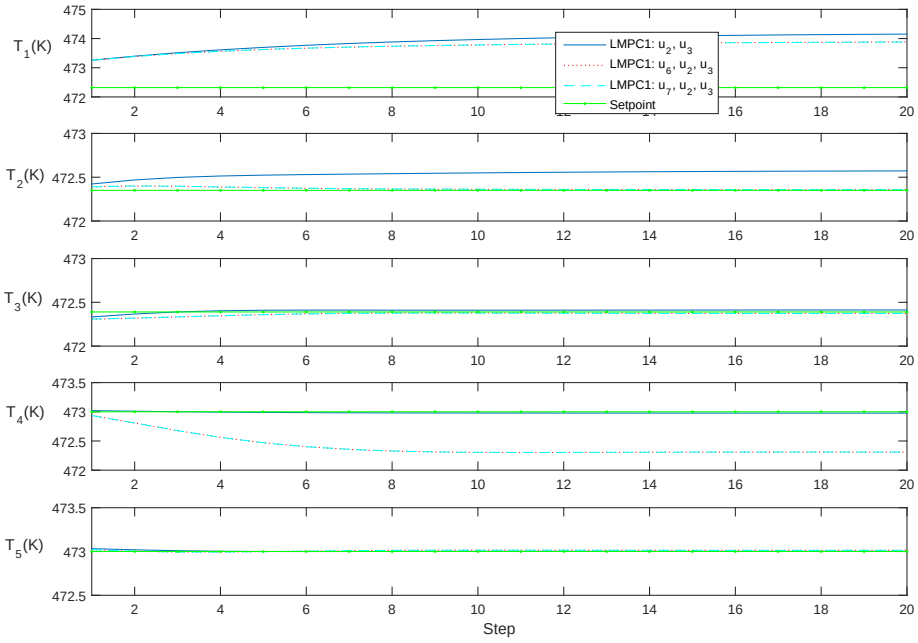

Fig. 9. The trajectories under performance optimization algorithm for the future 20 steps with current setpoint at time $t = 320s$ (sample 32)

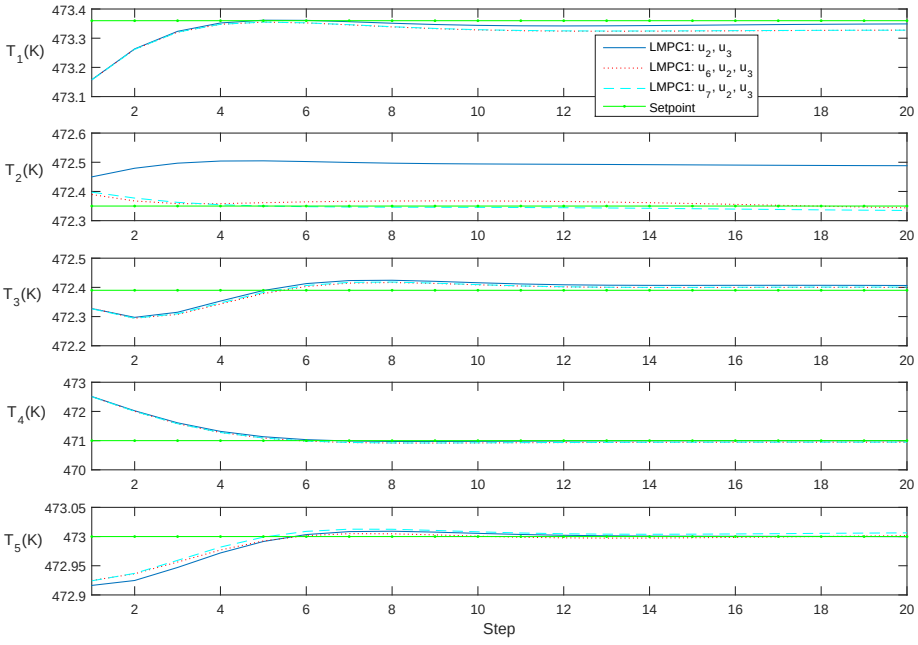

Fig. 10. The trajectories under performance optimization algorithm for the future 20 steps with newly designed setpoint at time $t = 320s$ (sample 32)

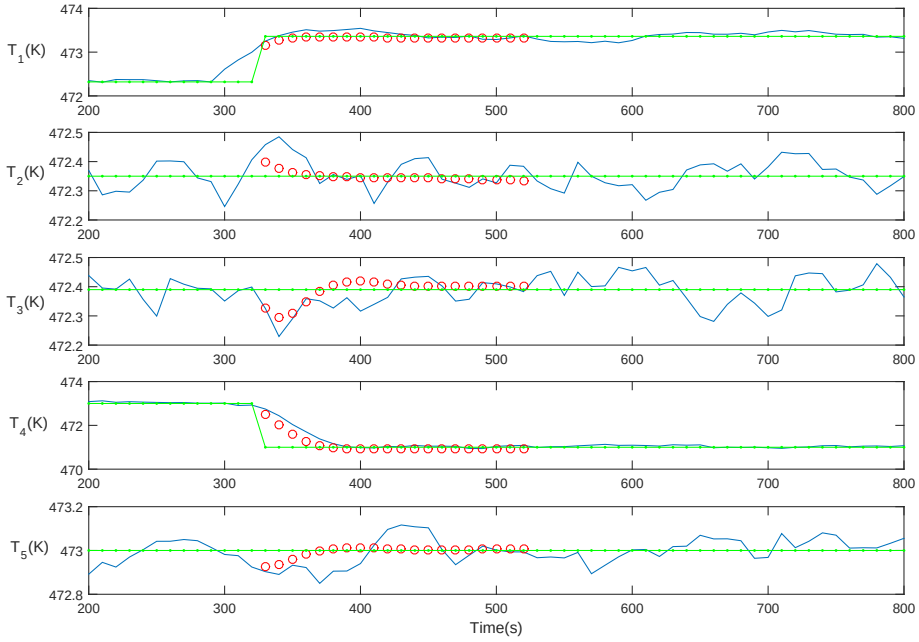

Fig. 11. Test results obtained by activating u_7 under the newly designed operating point (green dash-dot line: setpoint; blue solid line: temperature variations under DMPC; red circle line: trajectories under performance optimization algorithm)

Table 3
Computational time with the performance optimization algorithm and the DMPC

	POA	DMPC
Case study 1	2.89 s	37.8 s
Case study 2 (with current setpoint)	2.75 s	35.8 s
Case study 2 (with newly designed setpoint)	2.99 s	46.5 s

5.4.3 A note on the computational burden

The computation time for the performance optimization algorithm (POA) and for the DMPC, recorded by using the appropriate MATLAB tool, is reported in Table 3. Through the comparison, it can be easily noticed that the proposed method greatly reduces the computation time, which makes it suitable for on-line use.

6 Conclusions

This paper presents a performance optimization algorithm for controller reconfiguration in FTDMPC of large-scale systems. The performance optimization algorithm aims to check the ability and performance of the candidate reconfigured controllers in driving the process variables to the newly defined operating conditions. Under the assumption that the active constraints in non-faulty systems remain the same as they are at the nominal operating conditions, the global DMPC is split into two subproblems, which achieves the objective of rendering the computational burden compatible with on-line processing. The efficacy of the proposed performance optimization algorithm for controller reconfiguration has been demonstrated with two case studies on the alkylation of benzene process. Indeed, among the candidate reconfigurations, a non-square MPC design can be considered, which is achieved by adding more than one actuator in the place of a single, excluded faulty one. Even though the square MPC design is a little bit more common in practice, the use of additional actuators in the reconfiguration can be justified by the fact of introducing additional control capacities for fault compensation.

References

Alvarado, I., Limon, D., Muñoz de la Peña, D., Maestre, J., Ridao, M., Scheu, H., Marquardt, W., Negenborn, R., De Schutter, B., Valencia, F., Espinosa, J., 2011. A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. Journal of Process Control 21 (5), 800–815.

Blanke, M., Izadi-Zamanabadi, R., Bøgh, S., Lunau, C., 1997. Fault-tolerant control systems — A holistic view. Control Engineering Practice 5 (5), 693–702.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press, Cambridge, UK.

Camponogara, E., Jia, D., Krogh, B., Talukdar, S., 2002. Distributed model predictive control. IEEE Control Systems Magazine 22 (1), 44–52.

Chilin, D., Liu, J., Chen, X., Christofides, P., 2012a. Fault detection and isolation and fault tolerant control of a catalytic alkylation of benzene process. Chemical Engineering Science 78, 155–166.

Chilin, D., Liu, J., Davis, J., Christofides, P., 2012b. Data-based monitoring and reconfiguration of a distributed model predictive control system. International Journal of Robust and Nonlinear Control 22 (1), 68–88.

Chilin, D., Liu, J., de la Peña, D., Christofides, P., Davis, J., 2010. Detection, isolation and handling of actuator faults in distributed model predictive control systems. Journal of Process Control 20 (9), 1059–1075.

Farina, M., Betti, G., Giulioni, L., Scattolini, R., 2014. An approach to

distributed predictive control for tracking-theory and applications. IEEE Transactions on Control Systems Technology 22 (4), 1558–1566.
Gandhi, R., Mhaskar, P., 2008. Safe-parking of nonlinear process systems. Computers and Chemical Engineering 32 (9), 2113–2122.
Gandhi, R., Mhaskar, P., 2009. A safe-parking framework for plant-wide fault-tolerant control. Chemical Engineering Science 64 (13), 3060–3071.
Gani, A., Mhaskar, P., Christofides, P., 2007. Fault-tolerant control of a polyethylene reactor. Journal of Process Control 17 (5), 439–451.
Jämsä-Jounela, S.-L., Tikkala, V.-M., Zakharov, A., Pozo Garcia, O., Laavi, H., Myller, T., Kulomaa, T., Hmlinen, V., 2013. Outline of a fault diagnosis system for a large-scale board machine. International Journal of Advanced Manufacturing Technology 65 (9-12), 1741–1755.
Kettunen, M., Jämsä-Jounela, S.-L., 2011. Data-based, fault-tolerant model predictive control of a complex industrial dearomatization process. Industrial and Engineering Chemistry Research 50 (11), 6755–6768.
Kettunen, M., Zhang, P., Jämsä-Jounela, S.-L., 2008. An embedded fault detection, isolation and accommodation system in a model predictive controller for an industrial benchmark process. Computers and Chemical Engineering 32 (12), 2966–2985.
Liu, J., Chen, X., De la Peña, D., Christofides, P., 2010. Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. AIChE Journal 56 (8), 2137–2149.
Liu, J., De la Peña, D., Christofides, P., 2009. Distributed model predictive control of nonlinear process systems. AIChE Journal 55 (5), 1171–1184.
Luppi, P., Outbib, R., Basualdo, M., 2015. Nominal controller design based on decentralized integral controllability in the framework of reconfigurable fault-tolerant structures. Industrial and Engineering Chemistry Research 54 (4), 1301–1312.
Maciejowski, J., 1999. Modelling and predictive control: Enabling technologies for reconfiguration. Annual Reviews in Control 23, 13–23.
Maestre, J., Muoz De La Pea, D., Camacho, E., 2011. Distributed model predictive control based on a cooperative game. Optimal Control Applications and Methods 32 (2), 153–176.
Mahmoud, M., Jiang, J., Zhang, Y., 2003. Active fault tolerant control systems: Stochastic analysis and synthesis. Vol. 287. Springer.
Marro, G., Prattichizzo, D., Zattoni, E., 2003. A nested computational approach to the discrete-time finite-horizon LQ control problem. SIAM Journal on Control and Optimization 42 (3), 1002–1012.
Negenborn, R., De Schutter, B., Hellendoorn, J., 2008. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. Engineering Applications of Artificial Intelligence 21 (3), 353–366.
Negenborn, R., Maestre, J., 2014. Distributed model predictive control: An overview and roadmap of future research opportunities. IEEE Control Systems 34 (4), 87–97.
Prakash, J., Patwardhan, S., Narasimhan, S., 2002. A supervisory approach

to fault-tolerant control of linear multivariable systems. Industrial and Engineering Chemistry Research 41 (9), 2270–2281.

Pranatyasto, T., Qin, S., 2001. Sensor validation and process fault diagnosis for FCC units under MPC feedback. Control Engineering Practice 9 (8), 877–888.

Qin, S., Badgwell, T., 2003. A survey of industrial model predictive control technology. Control Engineering Practice 11 (7), 733–764.

Scattolini, R., 2009. Architectures for distributed and hierarchical model predictive control – A review. Journal of Process Control 19 (5), 723–731.

Scheu, H., Marquardt, W., 2011. Sensitivity-based coordination in distributed model predictive control. Journal of Process Control 21 (5), 715–728.

Skogestad, S., 2004. Control structure design for complete chemical plants. Computers & Chemical Engineering 28 (1), 219–234.

Sourander, M., Vermasvuori, M., Sauter, D., Liikala, T., Jämsä-Jounela, S.-L., 2009. Fault tolerant control for a dearomatisation process. Journal of Process Control 19 (7), 1091–1102.

Tao, J., Zhu, Y., Fan, Q., 2014. Improved state space model predictive control design for linear systems with partial actuator failure. Industrial and Engineering Chemistry Research 53 (9), 3578–3586.

Vahid Naghavi, S., Safavi, A. A., Kazerooni, M., 2014. Decentralized fault tolerant model predictive control of discrete-time interconnected nonlinear systems. Journal of the Franklin Institute 351 (3), 1644–1656.

Wang, L., Chen, X., Gao, F., 2013a. An LMI method to robust iterative learning fault-tolerant guaranteed cost control for batch processes. Chinese Journal of Chemical Engineering 21 (4), 401–411.

Wang, L., Mo, S., Zhou, D., Gao, F., Chen, X., 2012. Robust delay dependent iterative learning fault-tolerant control for batch processes with state delay and actuator failures. Journal of Process Control 22 (7), 1273–1286.

Wang, L., Mo, S., Zhou, D., Gao, F., Chen, X., 2013b. Delay-range-dependent method for iterative learning fault-tolerant guaranteed cost control for batch processes. Industrial and Engineering Chemistry Research 52 (7), 2661–2671.

Zattoni, E., 2008. Structural invariant subspaces of singular Hamiltonian systems and nonrecursive solutions of finite-horizon optimal control problems. IEEE Transactions on Automatic Control 53 (5), 1279–1284.

Zhang, R., Gan, L., Lu, J., Gao, F., 2013. New design of state space linear quadratic fault-tolerant tracking control for batch processes with partial actuator failure. Industrial and Engineering Chemistry Research 52 (46), 16294–16300.

Zhang, R., Lu, J., Qu, H., Gao, F., 2014a. State space model predictive fault-tolerant control for batch processes with partial actuator failure. Journal of Process Control 24 (5), 613–620.

Zhang, R., Lu, R., Xue, A., Gao, F., 2014b. Predictive functional control for linear systems under partial actuator faults and application on an injection molding batch process. Industrial and Engineering Chemistry Research

53 (2), 723–731.

Zhang, Y., Jiang, J., 2008. Bibliographical review on reconfigurable fault-tolerant control systems. Annual Reviews in Control 32 (2), 229–252.

Zheng, Y., Li, S., Li, N., 2011. Distributed model predictive control over network information exchange for large-scale systems. Control Engineering Practice 19 (7), 757–769.

Zheng, Y., Li, S., Wang, X., 2009. Distributed model predictive control for plant-wide hot-rolled strip laminar cooling process. Journal of Process Control 19 (9), 1427–1437.