Model Based Plug and Play Process Control of a Supermarket Refrigeration System: A Heuristic Approach

Axel G. Michelsen, Jakob Stoustrup and Roozbeh Izadi-Zamanabadi

Abstract— **In this paper, algorithms for automatic controller synthesis for a flexible, hierarchical system, which allows a model-based control system to be reconfigured when a sensor, an actuator or an entire new subsystem is plugged into a controlled process, are considered. The research reported here focuses on changing only part of a control system, keeping existing, well trusted controllers unless a substantial gain can be achieved. The developed method is successfully tested on an industrial case study from Danfoss A/S, where the process to be controlled is a flexible supermarket refrigeration system.**

I. Introduction

A complex process, such as a power plant or a water distribution system, might comprise hundreds or thousands of sensors and actuators.

Adding or removing just one sensor or actuator, however, will often require a redesign of the control system. Therefore, such changes are primarily implemented during a scheduled recommissioning of the process control system even though online reconfiguration would have yielded a more optimal performance. The lack of flexibility in such a system and the expenses involved with reconfiguration make the industry reluctant to implement advanced control technology in the first place or even upgrade the subsystems, for instance by adding sensors or actuators, in order to achieve optimal performance.

Traditionally, the high cost of controller design has been lowered by using PID controllers, and tuning these using heuristic tuning rules. See e.g. [1], [2]. This makes PID control the most commonly used controllers in industrial process control, because of the simple structure and ease of understanding it.

The reluctance towards using model based control technology might in part be ascribed to the expenses involved with recommisioning, even though, once the model based control system is operational it would yield a better performance.

It would be desired that new hardware, e.g. a new actuator or sensor, could be integrated in a process in a plug and play fashion, i.e., the controller automatically recognises that new hardware has been added to the process, and, using reliable nummerical methods, as a result reconfigures itself to accomodate theese changes, thus reducing or even removing the load on the designer.

This work is supported by The Danish Research Council for Technology and Production Sciences.

A.G. Michelsen and J. Soustrup are with Department of Electronic Systems, Aalborg University, Aalborg, Denmark (e-mail: {agm, jakob}@es.aau.dk)

R. Izadi-Zamanabadi is with Danfoss RA Advanced Engineering, Danfoss A/S, Nordborg, Denmark (e-mail: roozbeh@danfoss.dk)

A problem here is that the majority of the existing design methodologies are monolithic, i.e., given an open loop model of a process to be controlled they output a single mulitvariable controller.

Drastical changes to a control system, such as implementing a new, single controller when a new piece of hardware has been introduced, are not desireable, since it might be dificult to merge the new controller with the existing software, and the new behaviour of the controlled process might differ significantly from the old behaviour.

Plug and Play Process Control aims at lowering the cost associated with reconfiguring a process by automatically synthesizing new controllers after a process has been reconfigured. See e.g. [3], [4], [5]. This should not be confused with flexible manufacturing systems, where the purpose is to have a single manufacturing system that can manufature many different types of goods, see e.g. [6] for a survey. The purpose in plug and play process control is to have a one controller that is flexible regarding the individual subsystems, sensors and actuators of the process to be controlled.

This paper aims at designing plug and play algorithms that reconfigure a model based control system in localized manner, i.e., unless large savings are achievable, the controlled system should only behave different close to where new hardware has been installed. In order to do this the process to be controlled is divided into a hierachical system, such that localized changes to the controller can be made. This approach has previously been used for solving difficult problems, see e.g. [7], [8] and [9].

II. Problem Formulation

The system is described as a four touple, given as

$$H = (V, E, G, K), \tag{1}$$

where

V	is a set of vertices
E	is a set of edges
G	is a set of linear dynamical systems
K	is a set of linear dynamical controllers

The hierarchy of the system is described by a directed tree $T = (V, E)$, with exactly one sink as shown in Figure 1, where each vertice $v \in V$ corresponds to a linear system $G_v(s) \in G$, as seen in Figure 2 and controller $K_v(s) \in K$, and each edge $uv \in E$ corresponds to communication signals between system $G_u(s)$ and $G_v(s)$ denoted by ζ_{uv} and υ_{uv}.

The signals between $G_v(s)$, $K_v(s)$, parent system $G_w(s)$, and child systems $G_{u_1}(s)$, ..., $G_{u_N}(s)$, where N is the

ISBN 978-963-311-369-1

$$G_v(s) = \begin{bmatrix} G_{y_v u_v}(s) & G_{y_v \upsilon_{vw}}(s) & G_{y_v \zeta_{u_1 v}}(s) & \cdots & G_{y_v \zeta_{u_N v}}(s) \\ G_{\zeta_{vw} u_v}(s) & G_{\zeta_{vw} \upsilon_{vw}}(s) & G_{\zeta_{vw} \zeta_{u_1 v}}(s) & \cdots & G_{\zeta_{vw} \zeta_{u_N v}}(s) \\ G_{\upsilon_{u_1 v} u_v}(s) & G_{\upsilon_{u_1 v} \upsilon_{vw}}(s) & G_{\upsilon_{u_1 v} \zeta_{u_1 v}}(s) & \cdots & G_{\upsilon_{u_1 v} \zeta_{u_N v}}(s) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{\upsilon_{u_N v} u_v}(s) & G_{\upsilon_{u_N v} \upsilon_{vw}}(s) & G_{\upsilon_{u_N v} \zeta_{u_1 v}}(s) & \cdots & G_{\upsilon_{u_N v} \zeta_{u_N v}}(s) \end{bmatrix} \quad (2)$$

Fig. 2. The linear system $G_v(s)$ corresponding to a vertex $v \in V$.

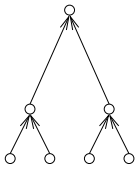

Fig. 1. A directed tree with exactly one sink used to describe the system hirarchy.

indegree of v, i.e., the number of child systems, are shown in Figure 3.

The signals corresponding to each pair of system and controller $(G_v(s), K_v(s))$ are,

- $u_v \in \mathcal{R}^{n_{u_v}}$, the local control signal from $K_v(s)$,
- $y_v \in \mathcal{R}^{n_{y_v}}$, the local measurements from $G_v(s)$,
- $Z_{p,v} = \{\zeta_{vw} \in \mathcal{R}^{n_{\zeta_{vw}}} : vw \in E\}$, a set of signals from $G_v(s)$ to parent system $G_w(s)$,
- $\Upsilon_{p,v} = \{\upsilon_{vw} \in \mathcal{R}^{n_{\upsilon_{vw}}} : vw \in E\}$, a set of signals from parent system $G_w(s)$ to $G_v(s)$,
- $Z_{c,v} = \{\zeta_{uv} \in \mathcal{R}^{n_{\zeta_{uv}}} : uv \in E\}$, a set of signals from child system to $G_v(s)$, and
- $\Upsilon_{c,v} = \{\upsilon_{uv} \in \mathcal{R}^{n_{\upsilon_{uv}}} : uv \in E\}$, a set of signals from G_v to child system.

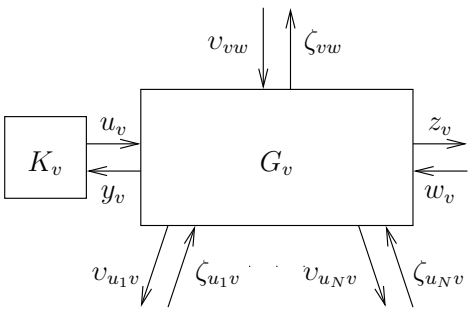

Fig. 3. The system, controller and communication signals associated to a vertex v with outdegree 1 and indegree N.

Remarks:

- The tree is directed in order to make it possible to distinguish between different time scales in the system.
- The sets $Z_{p,v}$ and $\upsilon_{p,v}$ are either the empty set if v is the sink, or contains exactly one signal otherwise.
- The sets $Z_{c,v}$ and $\upsilon_{c,v}$ are empty if v is a leaf, i.e., if there are no subsystems to G_v.
- In the special case where the tree describing the structure of the system has exactly one vertex v, G_v is similar to an ordinary two-port system since $Z_{p,v} = \Upsilon_{p,v} = Z_{c,v} = \Upsilon_{c,v} = \emptyset$.
- It is always possible to decompose a linear system as a hierarchical system, but it might not be prudent to do so if the system in question have strong couplings between the individual subsystems.

Using the above, the general problem is described; given a plug in of new hardware to the hierarchical system $H = (V, E, G, K)$ the local controllers $K_v \in K$, shall be reconfigured such that

$$\begin{aligned} \min_{U,\, K_U} \quad & \|U\| + f(H) \\ s.t. \quad & K_U = \{K_v \in K : v \in U\} \\ & U \subseteq V, \end{aligned} \quad (3)$$

where $\|U\|$ is the cost associated with reconfiguring the controllers in the K_U, and $f(H)$ is running cost of the process.

Some work has been done on characterising convex problems in decentralized control, see e.g. [10] or [11], but since it has been shown for similar problems, that there does not always exist a sequence of controllers of bounded order which obtains near-optimal control, nor an infinite-dimensional optimal controller [12], a heuristic approach for finding local controllers that will yield a suboptimal solution will be used instead of attacking the minimization problem directly.

III. Plug and Play Algorithms

Since finding a set of local controllers is in general intractable the local controllers will be synthesized one at a time. Furthermore, since synthesizing local controllers for all the subsets of vertices in a given tree will result in having to search through the power set of the vertices, a heuristic approach for choosing the local controller to be tuned is used. Two different heuristic approaches are given as algorithms in the following.

In the first approach it is assumed that subsystems on the same level are decoupled, and therefore, that there is no need for reconfiguring other controllers on the same level than the direct ancestors of where a change was made.

In the second approach it is assumed that changing a controller will require a reconfiguration of all the child

controllers. In both approaches it is assumed that the systems shows a high degree of locality, i.e., that changes to the control system will be most effective close to where the system was changed.

In the following, let $dist(v, u) : E \times E \rightarrow \mathcal{N}_0$ be the number of edges in the shortest path between v and u.

A. Algorithm 1

The first algorithm traverses a tree from the vertex where new hardware is plugged in, v_{in}, towards the sink, i.e., the controller to be synthesised will be the direct ancestors of the modified vertex. The algorithm will continue modifying controllers for ancestors vertexes until the cost of modifying a controller surpasses the gains achieved by modifying it.

Input: $H_{in} = (V_{in}, E_{in}, G_{in}, K_{in})$, $v_{in} \in V_{in}$
Output: $H = (V, E, G, K)$
Initialization: $v := v_{in}$
$H := H_{in}$
1: Synthesize new controller K_v for vertex v using models from H. Store result in H'.
2: if $\|v\| \geq f(H) - f(H')$ then return H
3: $H := H'$
4: if v is the sink then return H
5: $v :=$ parent of v
6: jump to 1

Algorithm 1

Since the algorithm returns the controller found during last iteration, T, if the cost of implementing the new controller, $||v||$, is greater than the gain in controller performance, $f(T)-f(T')$, the algorithm will not implement a new control law that is worse than the previous control law, i.e., the control performance will not degrade. Further more, since the algorithm traverses the hierarchy from the vertex where a change was made, v_{in}, towards the sink, choosing only the parent of the vertex, the algorithm has a worst case running time of $O(dist(v_{in}, sink))$, where $dist(v_{in}, sink)$ is the minimum number of edges from v_{in} to the sink.

B. Algorithm 2

The second algorithm also traverses a given tree from the vertex where new hardware is plugged in, but for each vertex where a new controller is synthesized all vertices lower in the hierarchy have a new controller synthesized as well.

The following helper function $S(H, v)$ synthesize new controllers for all vertices lower in the hierarchy than v.

Input: $H_{in} = (V_{in}, E_{in}, G_{in}, K_{in})$, $v_{in} \in V_{in}$
Output: $(H = (V, E, G, K), U)$
Initialization: $v := v_{in}$
$H := H_{in}$
$U := \{v_{in}\}$
1: Synthesize new controller K_v for vertex v using models from H. Store result in H.
2: for all children c of v
2.1: $(H, U') := S(H, c)$
2.2: $U := U \cup U'$
3: return (H, U)

Function: $S(H, v)$

Using the above function Algorithm 2 is constructed in the same way as Agorithm 1.

Input: $H_{in} = (V_{in}, E_{in}, G_{in}, K_{in})$, $v_{in} \in V_{in}$
Output: $H = (V, E, G, K)$
Initialization: $v := v_{in}$
$H := H_{in}$
1: $(H', U) := S(H, v)$
2: if $\|U\| \geq f(H) - f(H')$ then return H
3: $H := H'$
4: if v is the sink then return H
5: $v :=$ parent of v
6: jump to 1.

Algorithm 2

The argument about controller degradation given for Algorithm 1 also holds for Algorithm 2, and the outer algorithm has the same worst case running time. Since the helper function recursively synthesizes new controllers for all children of a given vertex it has a worst case running time of $O(n_V)$, where n_V is the number of vertices in the vertex set V_{in}. A conservative estimate of the worst case running time for Algorithm 2 is then $O(n_V dist(v_{in}, sink))$.

IV. Example

The above was used to reconfigure controllers for a linear approximation of a supermarket refrigeration system as shown in Figure 4.

In the supermarket refrigeration system liquid coolant is lead into an evaporator in the display case, where it is evaporated, thereby absorbing energy from the air in the display case. From the evaporator, the coolant is lead to the compressor where the pressure of the coolant is increas, such that it, in the condensor, can be condensed back into a liquid, and thereby release the stored energy to the surroundings. Air in the display case is ventilated past the goods, where it is heated by theese, thereby keeping them cold, and past the evaporator where it is cooled down again. The energy removed from the display case is controlled by the coolant valve, if more coolant is evaporated in the evaporator more energy is removed, and vice verse. A similar system has been used in [13] and presented as a benchmark for hybrid systems in [14].

In this example the focus is on the display case, so the compressor and the condensor are both ignored. A block diagram showing the structure of the linear approximation used in the following is shown in Figure 5.

Each piece of cooling furniture consists of a model of an evaporator, a model of the air temperature in the refrigerator, and a model of the goods temperature. The model of the evaporator, the model of the air temperature and the model of the goods temperature are found from a nonlinear model of the supermarket refrigeration system using system identification.

The found transfer functions are as follows:

$$\frac{T_{SH}(s)}{v_{pos}(s)} = \frac{-0.00665s - 7.786 \cdot 10^{-8}}{s^3 + 0.07379s^2 + 0.0003534s + 1.714 \cdot 10^{-11}} \quad (4)$$

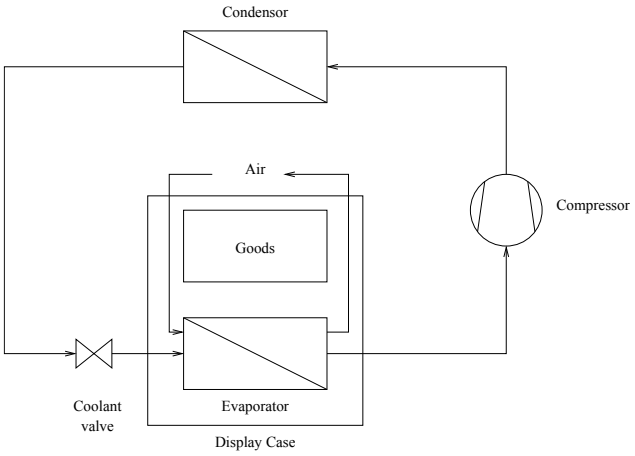

Fig. 4. Overview of the supermarket refrigeration system.

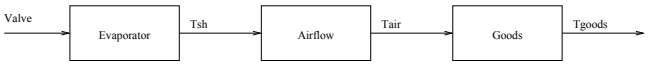

Fig. 5. Block diagram of the linear approximation of the supermarket refrigeration system.

$$\frac{T_{air}(s)}{T_{SH}(s)} = \frac{0.008448s + 9.762 \cdot 10^{-6}}{s^2 + 0.01693s + 1.927 \cdot 10^{-5}} \tag{5}$$

$$\frac{T_{goods}(s)}{T_{air}(s)} = \frac{0.0015}{s + 0.0015} \tag{6}$$

White noise is added to the valve position, the superheat temperature, the air temperature and the goods temperature. Initially, the variances are given as $v_v = .1$, $v_{sh} = 10$, $v_{T_{air}} = .01$ and $v_{T_{goods}} = .01$ respectably.

The hierarchy of a supermarket refrigeration system consisting of two display cases is shown in Figure 6. At the lowest level, i.e., the evaporator vertex, the super heat temperature of the evaporator is controlled. At the airflow vertex the temperature of the air circulating in the display case is controlled by sending super heat temperature references downwards, and at the top vertex the temperature of the goods is controlled by sending individual references to the air temperature to the airflow vertex.

A. Local Controller Synthesis and Control Structure

Each vertex in the hierarchy except the goods vertex has a local controller that is synthesized using the models and controllers of the vertices below it in the hierarchy. The parent of a given vertex is for controller synthesis purposes modelled as a constant reference, such that higher levels in the hierarchy sends references to lower levels.

The controllers are linear-quadratic-Gaussian controllers with a weight on the tracking error. The reference for the master controller is given as $T_{goods} = 5°C$. The resulting controller structure can be seen in Figure 7.

The weights for controllers synthesis are: $Q = 1, R = 1$ for the evaporator controller, $Q = 1, R = .01$ for the air temperature controller and $Q = 1, R = .1$ for the master

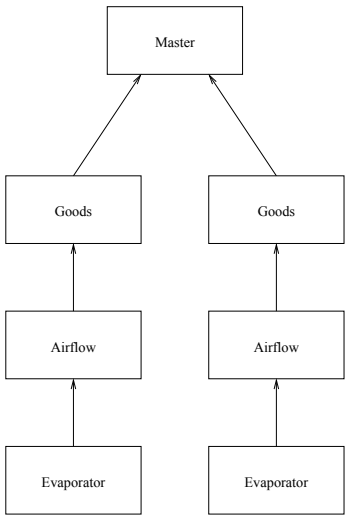

Fig. 6. Controller hierarchy of the supermarket refrigeration system.

controller, where Q is the weight of the reference error and R is the weight of the control signal sent to the child controller.

The cost for reconfiguring a controller is given as $5000 \cdot (dist(v, source) + 1)$, where $dist(v, source)$ is the distance from the vertex where the controller is located to the nearest source.

The cost for running the process, $f(H)$, is found by running a simulation of the system for one day and integrating the sum of the goods temperature tracking error squared and the valve position squared.

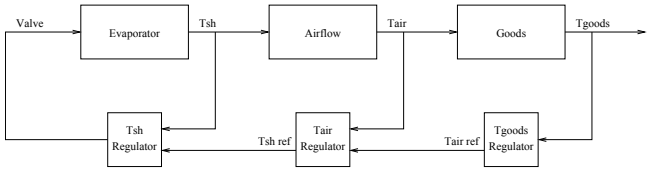

Fig. 7. Controller structure of the supermarket refrigeration system.

B. Plug and Play Scenarios

In the following plug and play scenario the evaporator of the display case is changed at time $T = 40,000s$, changing the variance of the superheat temperature from $v_{sh} = 10$ to $v_{sh} = 10$, resulting in a less noisy super heat temperature measurements, followed by a reconfiguration of the controllers using Algorithm 1. The plug and play scenario is shown in Figure 8.

Only Algorithm 1 is shown in this example, since, as a consequence of the local controller synthesis only uses models from lower levels to synthesise the local controller, Algorithms 1 and 2 will yield the same result.

After the change of the evaporator Algorithm 1 is run, resulting in a resynthesis of the evaporator controller.

The simulated super heat temperature, air temperature and goods temperature for the scenario are shown in Figure 9, 10 and 11. It is seen that the change of evaporator followed by a resynthesis of the evaporator controller only affects

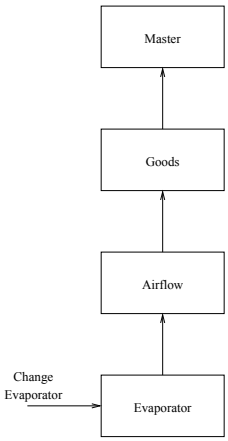

Fig. 8. Plug and play scenario: The evaporator in the display case is changed and the controllers are resynthesized.

the superheat temperature and the air temperature reference tracking.

A plot of the simulated running cost versus reconfiguration of the set of controllers for the system is shown in Figure 12. The plot confirms, that there is little to be gained by reconfiguring the air temperature controller and the goods temperature controller, after a plug in of a better evaporator has occured.

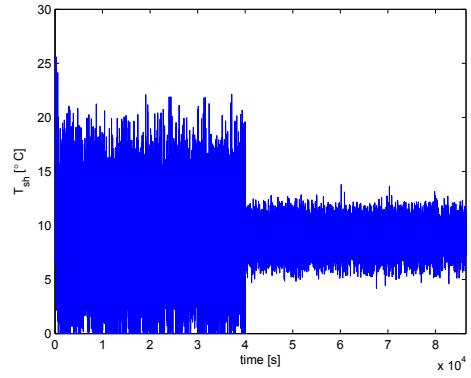

Fig. 9. Plot of the superheat temperature. At $T = 40,000$ a new superheat sensor is plugged in and the controller is recalculated.

V. Conclusion and Future Work

This paper presented a framework for hierarchical systems, in which it was possible to tune local model based controllers individually.

Using this framework, two algorithms for reconfiguring the local controllers after a new piece of hardware has been plugged in to a process was developed. One of theese developed algorithms was successfully tested on a case study provided by Danfoss A/S, where a new evaporator was plugged into an operating system and a new controller was synthesised on the fly.

In Figure 12 the cost of running the process is compared to the number of iterations of Algorithm 1. It seems that the

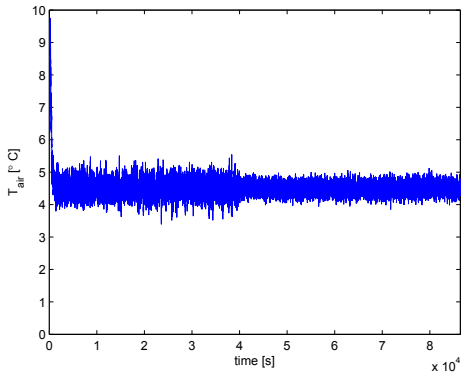

Fig. 10. Plot of the air temperature in the display case. At $T = 40,000$ a new superheat sensor is plugged in and the controller is recalculated.

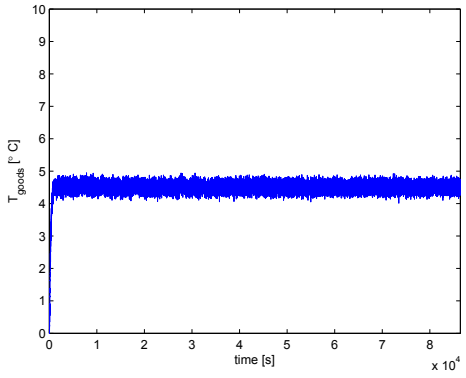

Fig. 11. Plot of the goods temperature in the display case. At $T = 40,000$ a new superheat sensor is plugged in and the controller is recalculated.

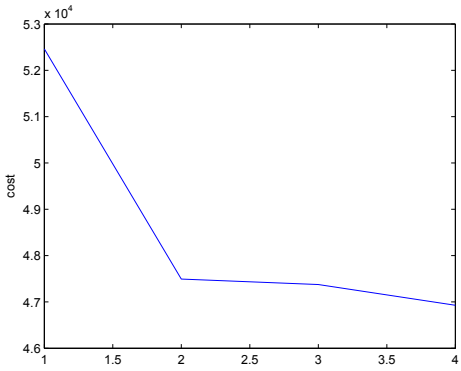

Fig. 12. Plot of the simulated running cost, $f(H)$, of the supermarket refrigeration system versus number of reconfigured controllers. 1 is the original system, 2 is the system with new evaporator and a new evaporation controller, 3 is as 2 with a new air temperature controller, and 4 is as with all controllers reconfigured.

heuristic approach of synthezising controllers close to where a change has been made is a sound approach. A prerequisite for the algorithm to work as favorably as shown is of course that the hierarchical model is well modelled, i.e., that the dynamics belonging to a vertex models the local dynamics, and not dynamics local to another vertice.

In future research two directions are considered. First is to change the way local controllers are synthesized, such that the two different algorithms will yield different results. Second is to develop algorithms that makes multible passes when synthesizing new local controllers.

REFERENCES

[1] J. Ziegler and N. Nichols, "Optimum settings for automatic controllers," *A.S.M.E*, vol. 64, pp. 759–768, 1942.

[2] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*. Wiley, 1996.

[3] A. Michelsen, R. Izadi-Zamanabadi, and J. Stoustrup, "Towards Automatic Model Based Controller Design for Reconfigurable Plants?" in *Proceedings of the 2008 IFAC World Congress, Seoul, Korea, July*, 2008.

[4] J. Bendtsen, K. Trangbaek, and J. Stoustrup, "Plug-and-Play Process Control: Improving Control Performance through Sensor Addition and Pre-filtering?" 2008.

[5] T. Knudsen, K. Trangbaek, and C. Kallesøe, "Plug and Play Process Control Applied to a District Heating System?" in *Proceedings of the 2008 IFAC World Congress, Seoul, Korea, July*, 2008.

[6] A. Sethi and S. Sethi, "Flexibility in manufacturing: A survey," *International Journal of Flexible Manufacturing Systems*, vol. 2, no. 4, pp. 289–328, 1990.

[7] P. Roberts and V. Becerra, "Optimal control of a class of discrete–continuous non-linear systems - decomposition and hierarchical structure," *Automatica*, vol. 37, no. 11, pp. 1757–1769, 2001.

[8] R. Scattolini and P. Colaneri, "Hierarchical model predictive control," in *Decision and Control, 2007 46th IEEE Conference on*, 2007, pp. 4803–4808.

[9] J. Raisch, A. Itigin, and T. Moor, "Hierarchical control of hybrid systems," in *Proc. 4th International Conference on Automation of Mixed Processes: Dynamic Hybrid Systems*, pp. 67–72.

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

[11] X. Qi, M. Salapaka, P. Voulgaris, and M. Khammash, "Structured optimal and robust control with multiple criteria: a convex solution," *Automatic Control, IEEE Transactions on*, vol. 49, no. 10, pp. 1623–1640, 2004.

[12] J. Stoustrup and H. Niemann, "Dynamical orders of decentralized controllers," *IMA Journal of Mathematical Control and Information*, vol. 16, pp. 299–308, 1999.

[13] L. Larsen, C. Thybo, and H. Rasmussen, "Potential Energy Savings Optimizing the Daily Operation of Refrigeration Systems."

[14] L. Larsen, R. Izadi-Zamanabadi, and R. Wisniewski, "Supermarket refrigeration system-benchmark for hybrid system control," Technical report, Aalborg University, http://www. control. aau. dk/hybrid, 2006, Tech. Rep.