Contents lists available at ScienceDirect

Journal of Process Control

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

Design, implementation and simulation of an MPC algorithm for switched nonlinear systems under combinatorial constraints

Adrian Bürger[a,b,*], Clemens Zeile[c], Angelika Altmann-Dieses[a], Sebastian Sager[c], Moritz Diehl[b,d]

[a] Faculty of Management Science and Engineering, Karlsruhe University of Applied Sciences, Moltkestraße 30, 76133 Karlsruhe, Germany
[b] Systems Control and Optimization Laboratory, Department of Microsystems Engineering (IMTEK), University of Freiburg, Georges-Koehler-Allee 102, 79110 Freiburg im Breisgau, Germany
[c] MathOpt Group, Institute for Mathematical Optimization, Faculty of Mathematics, Otto-von-Guericke University Magdeburg, Universitätsplatz 2, 39106 Magdeburg, Germany
[d] Department of Mathematics, University of Freiburg, Ernst-Zermelo-Straße 1, 79104 Freiburg im Breisgau, Germany

ARTICLE INFO

Article history:
Received 13 July 2018
Received in revised form 1 March 2019
Accepted 27 May 2019
Available online 28 June 2019

Keywords:
Model predictive control
Switched dynamic systems
Mixed-integer nonlinear programming
Optimal control
Approximation methods and heuristics

ABSTRACT

Within this work, we present a warm-started algorithm for Model Predictive Control (MPC) of switched nonlinear systems under combinatorial constraints based on Combinatorial Integral Approximation (CIA). To facilitate high-speed solutions, we introduce a preprocessing step for complexity reduction of CIA problems, and include this approach within a new toolbox for solution of CIA problems with special focus on MPC. The proposed algorithm is implemented and utilized within an MPC simulation study for a solar thermal climate system with nonlinear system behavior and uncertain operation conditions. The results are analyzed in terms of solution quality, constraint satisfaction and runtime of the solution steps, showing the applicability of the proposed algorithm and implementations.

1. Introduction

1.1. Motivation

The term *hybrid systems* refers to dynamic systems whose development is influenced by continuous as well as discrete variables [35]. A particular class of hybrid systems entitled *switched systems* refers to systems that are described by continuous states but which are subject to state-dependent discontinuities [9].[1] Such discontinuities can either be invoked internally depending on current system states, or by external influences which can be considered as controls of a system. Within this work, we will focus on the latter case.[2]

Switched systems can be described by a finite number of subsystems where the current activation status of a subsystem depends on the current system states and controls [30]. They can be found, e.g., in the process industry, the energy sector and in Heating, Ventilation and Air Conditioning (HVAC) of buildings, where systems often depend on both continuous controls (e.g. mass flow rates, valve positions) and discrete controls (e.g. operation status and modes of machinery). According systems and their components often show nonlinear behavior and are constrained to certain operation conditions, while additional constraints such as minimum operation time of machinery and start up costs have to be considered.

For control of systems that are generally subject to constraints and/or strongly depend on external factors (e.g. renewable energy systems), MPC yields great potential in terms of constraint-satisfactory and efficient system operation. However, application of MPC techniques leads to optimization problems that need to be solved on a time scale suitable for real-time control of these systems.

1.2. Relevant literature

If the number and sequence of switches on the considered control horizon is determined a priori for a system, switching time

* Corresponding author.
E-mail addresses: adrian.buerger@hs-karlsruhe.de (A. Bürger), clemens.zeile@ovgu.de (C. Zeile), angelika.altmann-dieses@hs-karlsruhe.de (A. Altmann-Dieses), sager@ovgu.de (S. Sager), moritz.diehl@imtek.uni-freiburg.de (M. Diehl).

[1] As mentioned in [38], such distinctions are not used thoroughly within the literature. Several examples can be found where the term hybrid system is used to describe systems that could be classified as switched systems, of which some are also referenced later within this work.
[2] Note that internal switches can also be formulated as external switches, cf. [9].

https://doi.org/10.1016/j.jprocont.2019.05.016

optimization can be applied in order to determine the optimal time points when the subsystems of the considered system need to be exchanged in order to follow a certain objective, without needing to actively consider discrete controls within the optimization problem. Applications and variants of this approach can be found, e.g., with [19,22,4,17]. However, application of this approach is problematic if the sequence and number of switches are not known a priori but have to be determined, cf. [39].

If active consideration of discrete controls cannot be avoided within the problem formulation, this leads to Mixed-Integer Optimal Control Problems (MIOCPs) that need to be solved on a suitable time scale for application within MPC of the system. For solving MIOCPs and Optimal Control Problems (OCPs) in general, the use of direct methods, especially direct multiple shooting [8] and direct collocation [47], is favorable [39,7]. For MIOCPs, this results either in Mixed-Integer Linear Programs (MILPs) or in Mixed-Integer Non-Linear Programs (MINLPs), depending, i.a., on the applied method, whether the considered dynamic system is linear(ized) or not, and on the type of constraints present within the problem formulation.

In the literature, several approaches for solution of mixed-integer programs within MPC applications are presented. For some applications, direct use of general solvers can be sufficient, as shown in [29] and [27] where corresponding MILPs are solvable within suitable time scales using CPLEX [24] and Gurobi [21], respectively. [5] presents an approach for modeling and predictive control of hybrid systems referred to as Mixed Logical Dynamical (MLD) systems (linear systems with both continuous and discrete inputs and states and subject to linear constraints) and solves the resulting MIOCPs using an available, general Mixed-Integer Quadratic Program (MIQP) solver.

Today, tailored algorithms that aim to exploit special aspects of MIOCPs for MLD systems have been presented, e.g., by [18]. This is exemplary for several approaches that can be found in the literature that aim towards decreasing solution times of mixed integer problems for real-time applications by tailored solution methods. [34] presents a tailored Branch-and-Bound (BnB) method incorporating a search space reduction to save computation time. [32] solves a mixed-integer MPC problem by application of a linear model-predictive controller with a longer prediction horizon in combination with a mixed-integer MPC problem with shorter horizon.

Compared to MILP, less work can yet be found regarding the utilization of MINLPs within real-time MPC. Since MINLPs are comparatively hard to solve, nonlinearities are in fact often avoided, e.g., by application of piecewise linearization for nonlinear functions, cf. [37]. However, such approximations might not always be applicable or sufficient. Apart from that, approaches can be found that aim towards faster solution of MINLPs by decomposing an original problem into a sequence of super- and subordinate optimization problems along different layers of abstraction of a problem, cf. [14], [44] and [33]. Possibilities for such separations are problem-specific and application can be problematic if abstraction layers cannot be sufficiently separated, e.g., if the allowance of operation of scheduled machinery depends on the actual system states that are subject to subordinate control levels.

A more general decomposition approach for fast solution of MINLP problems is presented by [39] where it is proposed to first solve a relaxed version of an MINLP, which is an Non-Linear Program (NLP), and obtain an integer trajectory for the discrete controls by application of a special rounding scheme entitled Sum-Up-Rounding (SUR). Though this might result in a suboptimal solution for the original problem, the error introduced by this approach is bounded and the method allows for solution of MINLPs within a time-scale that renders those real-time usable. Successful applications of this approach can be found in [28] and [15].

In [41] and more generalized in [53], an enhanced method for generating binary trajectories[3] named Combinatorial Integral Approximation (CIA) is proposed which can directly consider constraints such as Special Order Set 1 (SOS1)[4], maximum switching and dwell time constraints, but is more computationally demanding than SUR. A CIA problem is an MILP that can either be solved using a general MILP solver or by application of a fast, tailored BnB method. Additionally, [25] proposes to solve the MINLP once more subsequent to the binary approximation and with the obtained binary controls fixed, which again is an NLP, in order to adjust continuous controls and states to the obtained binary trajectories. This procedure yields the possibility to solve MIOCPs for nonlinear systems under quite arbitrary constraints, such as soft constraints for state boundaries, constraints on maximum switching and dwell times for discrete controls, as well as for specification of machinery operation conditions.

In [12], the applicability of the decomposition method described above for optimization-based control of a complex renewable energy system in the form of a Solar Thermal Climate System (STCS) is investigated by solving individual MIOCPs for a nonlinear model of the system which incorporates a single binary control variable and combinatorial constraints. It is shown that the solution time is tremendously decreased in comparison to using the general MINLP solver Bonmin [10] while comparable solution quality is achieved.

1.3. Contribution of this work

In this work, we build upon [12] and utilize the MINLP decomposition solution method within a warm-started algorithm for MPC of switched nonlinear systems under combinatorial constraints.

For this, we first present a new Python toolbox for solving CIA problems with special focus on application within MPC to facilitate easier use of the presented methods. Since the runtime of the CIA method rises with the number of binary variables treated, we introduce preprocessing heuristics for complexity reduction of CIA problems and show that those can decrease the solution time to a high extent for an exemplary benchmark problem.

Afterwards, we present a software implementation of the entire MPC algorithm and apply it within a simulated MPC loop for an extended version of the STCS model incorporating an increased number of states and continuous controls as well as two mutually exclusive binary control variables under dwell-time- and maximum-switching-constraints and whose operation conditions are subject to uncertainty. The performance of the algorithm and its implementation is assessed by an analysis of the obtained results in terms of solution quality, constraint satisfaction and runtime of the particular solution steps.

1.4. Notation

In the following, we denote by a_i the i-th element of a vector a. Further, we denote by A_i the i-th column of a matrix A and by $A_{(i, \ldots, j)}$ a submatrix of A that consists of all columns from i to j of this matrix. Indexing of vector and matrix elements starts at 0.

1.5. Used hardware and software

All computations are conducted on a Fujitsu P920 Desktop PC with an Intel Core i5-4570 3.20 GHz CPU and 32 GB RAM running

[3] Integer optimization problems can always be reformulated as binary optimization problems, cf. [39].

[4] The Special Order Set 1 (SOS1) condition causes that either only one or none of the binary controls of a problem can be active at any time.

Xubuntu 16.04 and with Python 2.7, gcc 5.4, CasADi 3.4.4, Ipopt 3.12.3. and SCIP 5.0.1.

2. Solution of MIOCPs for MPC applications

Within this section, we describe the decomposition approach for solution of MIOCPs utilized within this work as well as the CIA procedure in more detail. Further, we highlight selected aspects of problem formulation for MIOCPs within MPC.

2.1. Decomposition approach for approximate solution of MIOCPs

Let us consider a generic MIOCP with differential states $x(t) \in \mathbb{R}^{n_x}$, continuous controls $u(t) \in \mathbb{R}^{n_u}$, binary controls $b(t) \in \{0, 1\}^{n_b}$, slack variables $s(t) \in \mathbb{R}^{n_s}$ and time varying parameters $c(t) \in \mathbb{R}^{n_c}$ with $n_x, n_u, n_b, n_s, n_c \in \mathbb{N}$ on a given time horizon $t \in [t_0, t_f] \subset \mathbb{R}$ as in

$$\underset{x(\cdot),u(\cdot),b(\cdot),s(\cdot)}{\text{minimize}} \quad \int_{t_0}^{t_f} L(t, x(t), u(t), b(t), s(t)) \quad \mathrm{d}t + M(x(t_f)) \tag{1a}$$

$$\text{s. t.} \quad \text{for } t \in [t_0, t_f]:$$

$$\dot{x}(t) = f_0(x(t), u(t), c(t)) + \sum_{i=1}^{n_b} b_i(t) \cdot f_i(x(t), u(t), c(t)), \tag{1b}$$

$$r_{\text{lb}}(t) \le r(t, x(t), u(t), b(t), s(t), c(t)) \le r_{\text{ub}}(t), \tag{1c}$$

$$x(t_0) = x_0, \tag{1d}$$

$$x(t) \in \mathcal{X}, \quad u(t) \in \mathcal{U}, \quad b(t) \in \{0, 1\}^{n_b}. \tag{1e}$$

The objective of the (MIOCP) is to minimize the sum of a continuous-time Lagrangian cost functional $L(\cdot)$ and a Mayer term $M(\cdot)$. The Lagrangian term measures the "running" cost during the time horizon for given differential states, control and slack variables and is assumed to be integrable on the whole time horizon. The Mayer term, on the other hand, offers the option of including a desired differential state cost at the end of the time horizon in the objective. Note that we assume in (1b) a nonlinear system of explicit Ordinary Differential Equations (ODEs)[5], where the binary controls appear affinely so that we can write the right hand side as a sum of functions $f_0(\cdot)$ and $f_i(\cdot)$, $i = 1, \ldots, n_b$. The function $r(\cdot)$ contains path constraints with possibly time-dependent lower bound $r_{\text{lb}}(t)$ and upper bound $r_{\text{ub}}(t)$. Within $r(\cdot)$, different types of constraints, such as specifications of operation conditions for machinery, maximum switching constraints, dwell time constraints etc., can be specified. If desired, path constraints can also be formulated as soft constraints using elements of s. We assume that the functions $f_0(\cdot)$, $f_i(\cdot)$, $i = 1, \ldots, n_b$ and $r(\cdot)$ are continuously differentiable in all arguments within the domain of interest. By $\mathcal{X}$ and $\mathcal{U}$ we denote the feasible domains of the states and continuous controls, respectively.

For solution of OCPs, the use of direct methods (*first discretize, then optimize* approach) is favorable, especially direct multiple shooting [8] and direct collocation [47], which for MIOCPs results in MINLPs. For solving MINLPs within an MPC setting, we apply the decomposition approach proposed by [39] and solve first the relaxed MINLP with dropped binary control constraint, which is an NLP and therefore called NLP_{rel}. Next, we approximate the obtained relaxed binary controls $b_{\text{rel}} \in [0, 1]^{n_b}$ with binary controls $b_{\text{bin}} \in \{0, 1\}^{n_b}$ using CIA, which as described by [41] is an MILP. The final step is to solve the MINLP again with binary controls fixed, which again is an NLP and referred to as NLP_{bin}, in order to adjust the states x, continuous controls u and slack variables s to the obtained binary solution, cf. [25]. Algorithm 1 summarizes the steps of the decomposition approach.

Algorithm 1. Decomposition algorithm for solution of MIOCPs

```
   Input  : Discretized (MIOCP) instance with time grid $T_N$, initial guesses for $x, u, b$.
   Output: (Local) Optimal variables $x^*, u^*, b^*, s^*$ with objective $\mathcal{L}^* = \mathcal{L}(x^*, u^*, b^*, s^*)$.
1  Initialize $s = 0$ and solve $\text{NLP}_{\text{rel}} \rightarrow x, u, b_{\text{rel}}, s, \mathcal{L}_{\text{rel}}$.
2  if $\exists\, b_{\text{rel},i} \notin \{0, 1\} \mid b_{\text{rel},i} \in b_{\text{rel}}$ then
3      Solve (CIA) for $b_{\text{rel}} \rightarrow b_{\text{bin}}$;
4      Solve $\text{NLP}_{\text{bin}}$ with $b = b_{\text{bin}}$ fixed $\rightarrow x, u, s, \mathcal{L}_{\text{bin}}$;
5      return: $(x^*, u^*, b^*, s^*, \mathcal{L}^*) = (x, u, b_{\text{bin}}, s, \mathcal{L}_{\text{bin}})$;
6  else
7      return: $(x^*, u^*, b^*, s^*, \mathcal{L}^*) = (x, u, b_{\text{rel}}, s, \mathcal{L}_{\text{rel}})$;
```

Note that this approach does not solve the exact MINLP but an approximation given by a sequence of subproblems of which each one is less hard to solve than the original MINLP. Nevertheless, the approach is favorable for application here due to its real-time capabilities which we prefer due to our focus on MPC, while asymptotic objective error bounds for the approach depending on the grid step size have been proven [41]. In the upcoming section, the CIA method is described in more detail.

2.2. Combinatorial integral approximation under combinatorial constraints

The idea of CIA is to approximate the relaxed binary controls $b_{\text{rel}} \in [0, 1]^{n_b}$ with binary variables $b_{\text{bin}} \in \{0, 1\}^{n_b}$ in the sense that the maximum integrated difference of b_{rel} and b_{bin} for each discretization grid point is minimized. This distance is commonly measured with the maximum norm, cf. [25], and can be reformulated into an MILP which provides the opportunity to intuitively include combinatorial constraints. In the following, we are going to formally introduce this CIA problem.

Let the ordered set $\mathcal{G}_N = \{t_0 < \ldots < t_N = t_f\}$ denote a time grid with $\Delta t_i = t_{i+1} - t_i$ used for discretization of the controls in (MIOCP) meaning that the controls can only change values on these grid points $i = 0, \ldots, N-1$. Hence, the relaxed binary solution resulting from solving NLP_{rel} is in matrix form $b_{\text{rel}} \in [0, 1]^{n_b \times N}$. For formulating the dwell time constraints with minimum dwell times $L_k \in \mathbb{R}_+$, $k = 1, \ldots, n_b$, we introduce the grid index subsets $G_N^{i,L_k} := \{j = 1, \ldots, N-1 \mid t_j \in G_N\} \cap [t_i, t_i + L_k)$. By further introducing slack variables $\eta \in \mathbb{R}$, $s_{\sigma_{\max}} \in [-1, 1]^{n_b \times N-1}$, we define (CIA) in the presence of constraints on the maximum amount of switching actions per control $\sigma_{k,\max}$, $k = 1, \ldots, n_b$ on the given time horizon, and additional dwell time constraints, as follows:

$$\underset{\eta, b_{\text{bin}}, s_{\sigma_{\max}}}{\text{minimize}} \, \eta \tag{2a}$$

[5] Within this work, we only consider systems of explicit ODEs. Nevertheless, the approach can generally be extended for application for systems of implicit ODEs and Differential Algebraic Equations (DAEs).

$$\text{s. t. for} \quad k = 1, \ldots, n_b: \tag{2b}$$

$$\eta \geq \pm \sum_{j=0}^{i} \Delta t_j \cdot (b_{\text{rel},k,j} - b_{\text{bin},k,j}), \quad i = 0, \ldots, N-1, \tag{2c}$$

$$s_{\sigma_{\max},k,i} \geq b_{\text{bin},k,i} - b_{\text{bin},k,i-1}, \quad i = 1, \ldots, N-1, \tag{2d}$$

$$s_{\sigma_{\max},k,i} \geq -b_{\text{bin},k,i} + b_{\text{bin},k,i-1}, \quad i = 1, \ldots, N-1, \tag{2e}$$

$$\sigma_{\max,k} \geq \sum_{i=1}^{N-1} s_{\sigma_{\max},k,i}, \tag{2f}$$

$$b_{\text{bin},k,j} \geq b_{\text{bin},k,i} - b_{\text{bin},k,i-1}, \quad i = 1, \ldots, N-2, \quad j \in \mathcal{G}_N^{i,L_k}, \tag{2g}$$

$$1 - b_{\text{bin},k,j} \geq b_{\text{bin},k,i-1} - b_{\text{bin},k,i}, \quad i = 1, \ldots, N-2, \quad j \in \mathcal{G}_N^{i,L_k}, \tag{2h}$$

$$1 \geq \sum_{k=1}^{n_b} b_{\text{bin},k,i}, \quad i = 0, \ldots, N-1, \tag{2i}$$

$$b_{\text{bin},k,i} \in \{0, 1\}, \qquad i = 0, \ldots, N-1. \tag{2j}$$

Note that we minimize the accumulated difference between relaxed and binary control variables, as illustrated later in, e.g., Fig. 3, by adding constraint (2c). Since we are interested in the absolute value of this sum, we add the constraint both with plus and minus signs, indicated by "±". The inequality constraints (2d) to (2h) represent the combinatorial constraints described above, while (2i) is the so-called SOS1 condition, which causes exactly one mode or the off mode to be active at any time.

Overall, the solution of this problem yields a binary approximation $b_{\text{bin}} \in \{0, 1\}^{n_b \times N}$ for b_{rel} with an approximation error that is bounded depending on the discretization grid. Further details on CIA can be found in [41] and [25].

2.3. Time transformation of ODEs

If a time grid $\mathcal{G}_N$ used within a moving-horizon MPC application consists of non-equidistant time points, the duration of a certain grid interval Δt_i can change in between two MPC steps. To avoid regeneration of the problem discretization and the according derivatives needed by an NLP solver, a time transformation to the ODE right hand side can be applied, so that at time point t_i, instead of

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(t, \cdot), \quad t \in [t_i, t_{i+1}], \tag{3}$$

the term

$$\frac{\mathrm{d}x}{\mathrm{d}\tau} = \Delta t_i \cdot f(\tau, \cdot), \quad \tau \in [0, 1], \tag{4}$$

is used for formulation of the problem discretization, while the corresponding values of the time points t_i and t_{i+1} enter NLP_{rel} and NLP_{bin} as parameters. According formulations are also used within switching time optimization, where the durations of time steps enter an OCP as optimization variables, cf. [19].

2.4. Formulation of combinatorial constraints as smoothened vanishing constraints

If the allowance for activation of binary switches b within a mixed-integer optimization problem depends on conditions $h(\cdot)$ as in

$$0 \leq h(x(t), u(t), b(t), s(t), c(t)), \tag{5}$$

which is a particular case of the path constraints (1c), there exist several ways to formulate such conditions, while the choice of formulation can have different effects on the optimization problem. An overview and comparison of different formulation techniques in the context of this work is given by [26]. According to the results given there, we will in the following use smoothened vanishing constraints

$$-\epsilon \leq b(t) \cdot h(x(t), u(t), s(t), c(t)), \quad \text{with} \quad \epsilon \in \mathbb{R}_+, \quad \epsilon \to 0, \tag{6}$$

for formulation of combinatorial constraints to prevent, e.g., violation of Linear Independence Constraint Qualification (LICQ) conditions, cf. [26].

3. A new software package for solution of CIA problems

In the following, we introduce a new software package for solution of CIA problems including preprocessing heuristics for complexity reduction of this problem class.

3.1. Introduction of the software package pycombina

According to [41], (CIA) can either be solved using a general MILP solver such as Gurobi or SCIP [20], or a BnB algorithm tailored to the structure of (CIA) can be applied which was found to be more efficient in terms of computation time. [25] presents several variants of the BnB algorithm further referred to as (BnB).

To simplify the use of methods and algorithms presented within this work, we introduce the open-source software package *pycombina* for solution of (CIA). With *pycombina*, we provide a Python module for solving (CIA) under consideration of combinatorial constraints. *pycombina* is licensed under the GNU Lesser General Public License (LGPL) and contains a fast implementation of (BnB) presented in [41] and [25] using a *best-first* search strategy, i.e., the first solution found by the method is also a global optimum of (CIA). While major parts of *pycombina* such as the problem setup and preprocessing heuristics (see Section 3.2) are written in Python to favor flexible applicability and extensibility, computationally heavy parts of (BnB) are implemented in C++ and interfaced using *pybind11* [50] to increase computational performance. Maximum-switching-, SOS1- and dwell-time-constraints are, if present in a problem, actively considered within the implementation of (BnB) to allow for taking off bigger steps and shortcuts, cf. [25]. In addition to (BnB), (CIA) can be solved using Gurobi and SCIP, which is mainly intended for benchmarking purposes.[6]

3.2. A complexity reduction heuristic for CIA problems

Since the effort to solve (CIA) can increase strongly with the number of variables considered for the problem, we propose preprocessing heuristics for complexity reduction of CIA problems. The main part of the procedure is based on the idea that, if already in the relaxed solution a binary variable is constantly 0 or 1 (or sufficiently close to one of these values) over several time steps, then this is likely to persist for the corresponding time points in the binary solution. Therefore, the CIA solution method does not need to process every time point in this case, but can handle several time points of the grid within bigger time steps. Since it is likely that many relaxed binary values will already be close to either 0 or 1 due to the typical bang-bang-type solution of OCPs [39], grouping of such sequences can be expected to reduce the number of time points that need to be processed to a high extent.

A detailed description of the procedure is given in Algorithm 2. Iterating over the relaxed binary solution given in matrix form, submatrices of a column span δ are checked whether all elements

[6] For solving (CIA) with Gurobi or SCIP within *pycombina*, the user needs to provide adequate licenses for these solvers. (BnB) however, which is the recommended solution method within *pycombina*, as well as *pycombina* itself can be used without additional licenses.

are sufficiently close to either 0 or 1 depending on a threshold ε and whether all columns of such a submatrix are identical. If this is the case, the center column of this submatrix is marked not to be considered within the reduced problem. The smaller the column span parameter δ is chosen, the more columns will potentially be removed, but also the more degrees of freedom might be taken away from the CIA solution method.

Algorithm 2. Complexity reduction heuristic for CIA problems

```
Input     : Time grid $\mathcal{G}_N$, relaxed binary solution $b_{\text{rel}} \in [0,1]^{n_b \times (|\mathcal{G}_N|-1)}$.
Output    : Reduced time grid $\mathcal{G}_{N,\text{red}}$ with $|\mathcal{G}_{N,\text{red}}| \leq |\mathcal{G}_N|$, reduced relaxed binary
            solution $b_{\text{rel,red}} \in [0,1]^{n_b \times (|\mathcal{G}_{N,\text{red}}|-1)}$.
Parameters: Submatrix column span $\delta \in \mathbb{N} \geq 3$, binary value threshold $\varepsilon \in \mathbb{R}_+$.
1  Initialize mask $\iota \leftarrow 1^{(|\mathcal{G}_N|-1)}$, empty vector $\phi \leftarrow ()$;
2  for each $b_k \in [0,1]$ in $b_{\text{rel}}$ do
3      // Relaxed binary values with $b_k \leq \varepsilon$ and $b_k \geq 1-\varepsilon$ are considered binary
4      if $b_k \leq \varepsilon$ then
5          $b_k \leftarrow 0$
6      if $b_k \geq 1-\varepsilon$ then
7          $b_k \leftarrow 1$
8  for $i \leftarrow 0$ to $(|\mathcal{G}_N| - \delta - 2)$ by 1 do
9      // Select a submatrix of $\delta$ columns
10     $b^i_{\text{rel}} \leftarrow b_{\text{rel},(\text{i},\ldots,\text{i}+\delta)}$;
11     // Check if all elements of the selected submatrix are binary
12     if $b^i_{\text{rel},j} \in \{0,1\}^{n_b}\ \forall\ b^i_{\text{rel},j} \in b^i_{\text{rel}}$ then
13         // Check if all columns of the selected submatrix are identical
14         if $b^i_{\text{rel},j} = b^i_{\text{rel},k}\ \forall\ k = 2,\ldots,\delta$ then
15             // Mark the center column not to be considered in the reduced problem
16             $\iota_{\lfloor \frac{\delta}{2} \rfloor} \leftarrow 0$;
17 for $i \leftarrow 0$ to $(|\mathcal{G}_N| - 2)$ by 1 do
18     if $\iota_i = 1$ then
19         // Append $i$ at the end of vector $\phi$
20         $\phi \leftarrow \phi || (i)$;
21 return: $(\mathcal{G}_{N,\text{red}}, b_{\text{rel,red}}) \leftarrow (\mathcal{G}_{N,\phi} || \mathcal{G}_{N,|\mathcal{G}_N|}, b_{\text{rel},\phi})$;
```

In addition to that, further measures have been identified and included in *pycombina* that can reduce the effort for solving (CIA) to a greater extent:

1 binaries whose relaxed solution is constantly 0 over the whole time horizon are directly set to 0 and not considered within the problem formulation;
2 in case a control is constantly 1 over the whole horizon, the solution is directly given since all other controls need to be equal to 0 due to the SOS1 constraints;
3 in case the required number of switches indicated by the relaxed solution for a binary control is less than the number of maximum switches defined by the user, it can be automatically reduced to this number.

3.3. *Performance benchmark for* pycombina

Within a brief benchmark, we compare the performance of the different solution methods incorporated in *pycombina* and the impact of the preprocessing heuristics on the solution time and quality of a problem.

3.3.1. *Benchmarking setup*

The input data used within this benchmark is the relaxed solution of the binary variables from the *Lotka Volterra Multimode fishing problem* taken from the MIOCP benchmark library `mintoc.de` [40] as shown in Fig. 1, which contains $n_b = 3$ binary control inputs that depict different fishing options. To investigate runtime development for an increasing number of optimization variables, we solve a CIA problem for data sets representing the trajectories in Fig. 1 by $N_1 = 60$, $N_2 = 120$, $N_3 = 240$ and $N_4 = 480$ data points on an equidistant time grid from $t_0 = 0$ to $t_f = 12$, once using (BnB) and once using SCIP. For each solver, we solve the problem once in original size and once after application of the preprocessing heuristics. For this study, we determine a maximum of $\sigma_{\max} = 2$ switching actions per control input and that at most one control can be active at a time.

3.3.2. *Runtime of solution methods and impact of the preprocessing heuristics*

Fig. 2 shows a comparison of the solution times obtained by application of the BnB algorithm and SCIP as well as the impact of the preprocessing heuristics. In the top graph of Fig. 2, it can be seen that (BnB) is capable of solving the same CIA problem up to several orders of magnitude faster than SCIP. While SCIP takes ≈170.6 s to

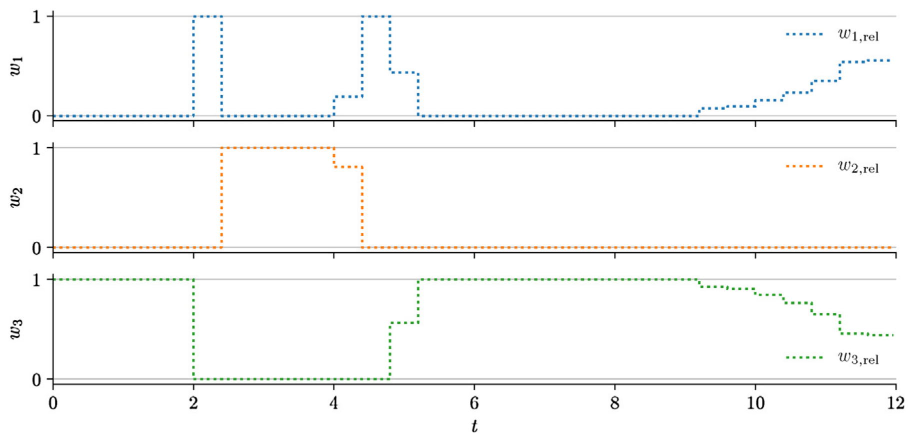

Fig. 1. Relaxed binary solution from the *Lotka Volterra Multimode fishing problem* [40].

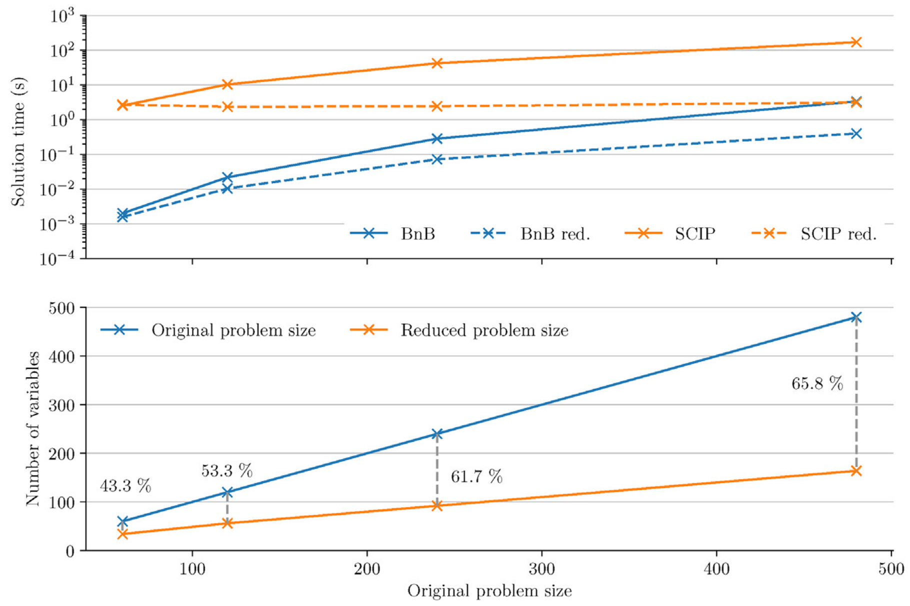

Fig. 2. The top graph shows the runtime of (BnB) and SCIP for the original problem instance (BnB, SCIP) and for the problems with reduced size (BnB red., SCIP red.). The bottom graph shows the number of time steps in the original and corresponding reduced problem as well as the relative reduction of steps.

solve the biggest problem instance N_4, (BnB) is able to solve the problem in ≈3.4 s.

The bottom graph of Fig. 2 shows that the preprocessing heuristic is able to reduce the number time steps of (CIA) that need to be processed within this benchmark to a high extent, from 43.3% for N_1 up to 65.8% for N_4. By doing this, the solution time for a problem can be reduced by a minimum of approximately one order of magnitude for this example for both solution methods, so that for the biggest problem instance N_4, the solution time of (BnB) is further reduced to ≈0.4 s.

3.3.3. *Uniqueness of solutions and optimality after preprocessing*

Exemplary, the solution of (CIA) for the data set represented by $N_3 = 240$ data points obtained by (BnB) is shown in Fig. 3. Both maximum-switching- and SOS1-constraints are fulfilled. However, it can be observed that the solutions of the original and the reduced problem with $N_{3,\mathrm{red}} = 92$ slightly differ.

On the one hand, this situation can occur if the global optimum of a CIA problem is not unique, i.e., there can exist several solutions that are able to yield the same objective value. In that case, the result is still a global optimum, while different binary trajectories are obtained due to different execution orders of the solution methods. On the other hand however, since the preprocessing heuristics remove degrees of freedom from (CIA), situations might also occur where the global optimum of the reduced problem is in fact different from the solution of the original problem, which cannot be obtained anymore due a reduction of degrees of freedom in the modified problem setup.

Within the scenario shown in Fig. 3 the latter holds true, and the obtained objective of the original problem is $\eta_3 = 0.4171$ and $\eta_{3,\mathrm{red}} = 0.534$ for the reduced problem. This reflects a trade-off between runtime reduction and optimality possibly introduced by the preprocessing heuristics and which can be influenced, i.a., by adjustment of the column span parameter δ shown in Algorithm 2.

4. Design and implementation of the MPC algorithm

Within this section, we describe the methods and software used for implementation of the MPC algorithm and highlight specific aspects. The overall procedure and the main stages of the implemented algorithm are illustrated in Algorithm 3.

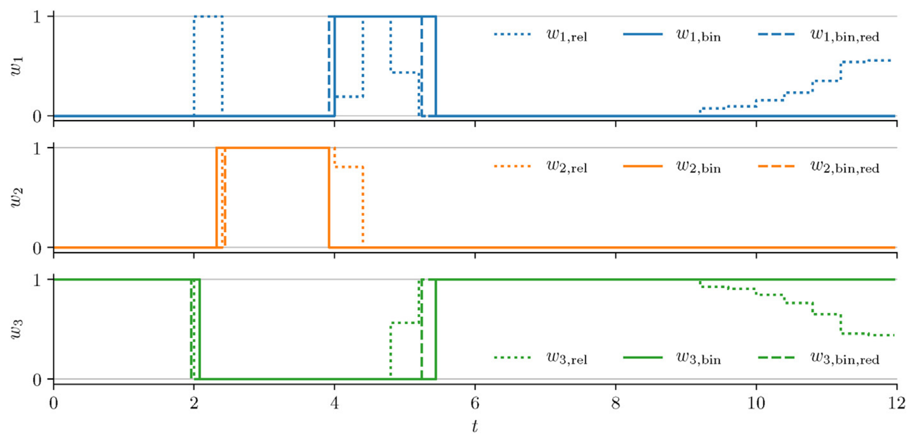

Fig. 3. Solutions of the original problem with $N_3 = 240$ and corresponding reduced problem with $N_{3,\mathrm{red}} = 92$ data points of the *Lotka Volterra Multimode fishing problem* obtained using (BnB).

4.1. Discretization of the MIOCP and utilized solvers

The discretization of the MIOCP is conducted using direct collocation with Lagrange polynomials with Radau collocation points of order 3 [6] and implemented using the open-source dynamic optimization framework CasADi [3] in Python. For solution of NLPs, we use Ipopt [49] with linear solver MA27 [23].[7] The CIA problem is solved using the *pycombina* software package presented in the previous section.

Algorithm 3. Main stages of the implemented MPC algorithm

```
   Input      : Discretized (MIOCP) instance with initial time grid $\mathcal{G}_N$, initial guess
                generator $G$, forecast source $F$, field interface $I$, user-defined stopping
                criterion $S$ (optional)
   Parameters: warm-start IPM barrier parameter $\mu_{\mathrm{ws}}$, final IPM barrier parameter $\mu_{\mathrm{f}}$
 1 Initialize timer $\tau \leftarrow \mathcal{G}_N^0$ and start timer;
 2 Obtain $c$ through $F$ and current system state $x_0$ through $I$;
 3 Obtain initial guess $x_{\mathrm{init}}, u_{\mathrm{init}}, b_{\mathrm{init}}$ given $c, x_0$ using $G$;
 4 while not $S$ fulfilled do
 5     Solve (NLP$_{\mathrm{rel}}$) online for $x_{\mathrm{init}}, u_{\mathrm{init}}, b_{\mathrm{init}}$ given $c, x_0$ until $\mu_{\mathrm{f}} \rightarrow x, u, b_{\mathrm{rel}}, s, \mathcal{L}_{\mathrm{rel}}$;
 6     Solve (CIA) online for $b_{\mathrm{rel}} \rightarrow b_{\mathrm{bin}}$;
 7     Solve (NLP$_{\mathrm{bin}}$) online for $x, u, s$ given $c, x_0$ and $b = b_{\mathrm{bin}}$ fixed until $\mu_{\mathrm{f}}$
         $\rightarrow x, u, b_{\mathrm{bin}}, s, \mathcal{L}_{\mathrm{bin}}$;
 8     Send $(x_{\mathrm{opt}}, u_{\mathrm{opt}}, b_{\mathrm{opt}}, s_{\mathrm{opt}}, \mathcal{L}_{\mathrm{opt}}) \leftarrow (x, u, b_{\mathrm{bin}}, s, \mathcal{L}_{\mathrm{bin}})$ through $I$;
 9     if (MIOCP) contains dwell time constraints then
10         Fix $b$ for next (MIOCP) for remaining dwell times;
11     Adjust $\mathcal{G}_N$ (step forward in time) and update $c$ through $F$ for next MPC step;
12     Solve (NLP$_{\mathrm{rel}}$) offline for $x, u, b_{\mathrm{bin}}, s$ given $c, x_0$ until $\mu_{\mathrm{ws}}$
         $\rightarrow x_{\mathrm{init}}, u_{\mathrm{init}}, b_{\mathrm{init}}, s_{\mathrm{init}}, \mathcal{L}_{\mathrm{rel,init}}$;
13     Pause until $\tau = \mathcal{G}_N^0$;
14     Obtain current system state $x_0$ through $I$;
15     if (MIOCP) contains dwell time constraints then
16         Fix $b$ for next (MIOCP) for remaining dwell times;
```

[7] Both academic and personal licenses of MA27 can be obtained free of charge from HSL. Nevertheless, if a pursued application does not meet the terms stated by [23], using the open-source linear solver MUMPS [1,2] within Ipopt can be considered.

4.2. Initial guess generation and warm-starting of subsequent MPC steps

To facilitate high performance in terms of solution quality and runtime of NLP solution methods, careful initialization of optimization variables is required.

For initialization of the first NLP within an MPC loop, initial guesses can be obtained, e.g., by simulation of a system $f_{\mathrm{C}}(\cdot)$ which, in addition to $f(\cdot)$, contains a conventional control scheme based on PID- and set-point-based control. We use Modelica to implement $f_{\mathrm{C}}(\cdot)$ and OpenModelica [46] for simulation of $f_{\mathrm{C}}(\cdot)$ via the OpenModelica Python interface OMPython [31].

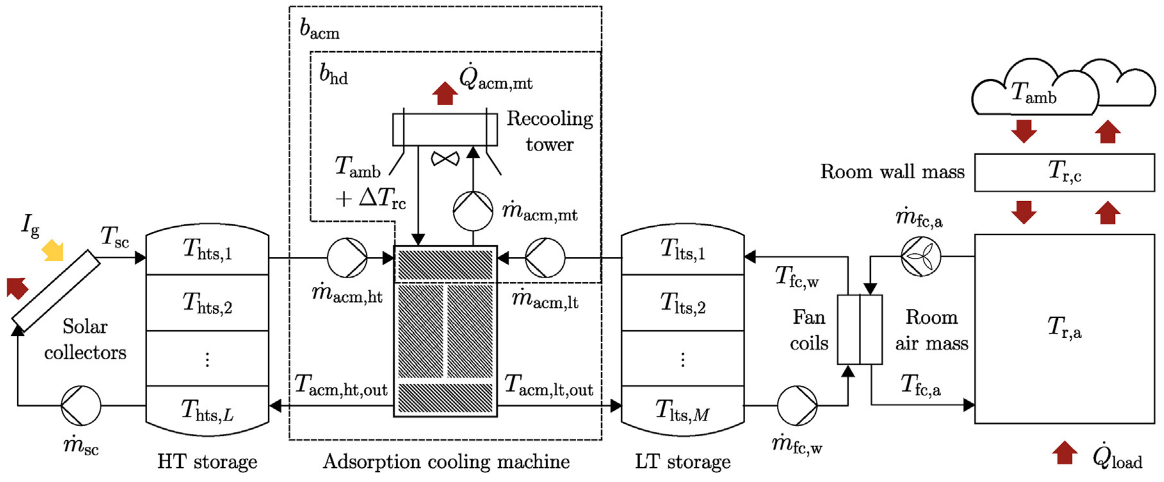

Fig. 4. Depiction of the STCS considered within this study. Red arrows indicate heat exchange, yellow arrows indicate solar irradiation.

For initialization of the next MPC solution step, it is particularly useful to re-use the results of a previous optimization cycle. If applied reasonably, *warm starting* of solvers can save valuable computation time within MPC applications. For warm starting of Interior Point Methods (IPMs), a methodology is presented by [52] which we partially adopt for warm-starting of Ipopt.

However, the solution of the second NLP with binaries fixed is not necessarily a good initialization for the first NLP of the next MPC step where binaries are both free and relaxed again, since the solution of the second NLP heavily relies on the results of the approximation of the binary controls. To overcome this issue, we first divide our MPC algorithm into an *online* part that needs to be executed as fast as possible right after new data for the system state has been obtained, and a subsequent *offline* part where more computation time is available. Within the online part of the algorithm, optimized system controls are obtained, while we use the offline part to determine an initialization for the next upcoming MPC step. To achieve that, once the online part has finished and optimized controls for the binary problem have been obtained, we start to solve the same NLP again in its relaxed form, but only until a certain barrier parameter is reached. Once new data for the system state and for the time-varying parameters has been obtained, we update x_0 and c and afterwards continue to solve the relaxed NLP, which is now parametrized for the new MPC step.

5. Simulation study for a solar thermal climate system

Within this section, the framework introduced within the previous sections is applied for the model of a Solar Thermal Climate System (STCS) within a simulation study. First, we give a detailed description of the considered system modeling and its dynamics, components, dependencies and operation conditions. Then we present the MIOCP for the system and the implementation and conduction of the simulation study. Finally, we analyze the obtained results in terms of solution quality, constraint satisfaction and runtime.

5.1. System description

The system considered for this study is a modified version of a climate system installed at the Faculty of Management Science and Engineering at Karlsruhe University of Applied Sciences. A schematic depiction of the system is given in Fig. 4.

The core component of the system is a silica-gel/water Adsorption Cooling Machine (ACM) which, when switched on ($b_{\mathrm{acm}} = 1$), utilizes the heat from a stratified High Temperature (HT) water storage of volume $V_{\mathrm{hts}} = 2.0\,\mathrm{m}^3$ to absorb heat from a stratified Low Temperature (LT) water storage with volume $V_{\mathrm{lts}} = 1.0\,\mathrm{m}^3$ and emits the combined heat through a recooling tower located on the roof of the building. In times of low ambient temperature and if the ACM is not in operation, the recooler can also be used autonomously in a free cooling or Heat Dissipation (HD) mode ($b_{\mathrm{hd}} = 1$) to cool down the LT storage directly at the ambient without use of the ACM.

The HT storage is supported by an array of horizontally placed solar thermal collectors which are connected to the system through a heat exchanger that separates the outdoor, frost-proof solar circuit from the indoor water circuits. The solar collectors have a total collector surface of $A_{\mathrm{sc}} = 35.0\,\mathrm{m}^2$, an optical efficiency of $\eta_{\mathrm{sc}} = 0.8$ and are exposed to the solar irradiation I_{g}. Regarding the mass flow rate $\dot{m}_{\mathrm{sc}}$ of the solar fluid through the collectors, we assume that the pump that produces the mass flow, if turned on, can be set on a continuous scale $\dot{m}_{\mathrm{sc}} \in [0.1, 0.5]\,\mathrm{kg/s}$.[8] The pump on the water side of the heat exchanger behaves identically with $\dot{m}_{\mathrm{hx}} \in [0.1, 0.5]\,\mathrm{kg/s}$. The heat exchanger has a volume of $V_{\mathrm{hx}} = 3.77 \times 10^{-3}\,\mathrm{m}^3$ on each side with heat exchange surface $A_{\mathrm{hx}} = 4.02\,\mathrm{m}^2$.

Cooled-down water can be extracted from the bottom of the LT storage to support a set of fan coils operated for heat exchange with the air mass $m_{\mathrm{r,a}} = 2.16 \times 10^3\,\mathrm{kg}$ of a room. The water mass flow rate $\dot{m}_{\mathrm{fc,w}}$ through the fan coils can be set on a continuous scale $\dot{m}_{\mathrm{fc,w}} \in [0.1, 0.3]\,\mathrm{kg/s}$ if the corresponding pump is turned on. For simplicity, we assume the air mass flowing through the fan coils to be constant at $\dot{m}_{\mathrm{fc,a}} = 0.43\,\mathrm{kg/s}$.[9]

In addition to the fan coils, the room temperature $T_{\mathrm{r,a}}$ is influenced by a time-varying heat load $\dot{Q}_{\mathrm{load}}$ that is usually due to influences from heating through solar irradiation on the building, air exchange, internal heat loads from machines and humans etc. The room air further exchanges heat with the concrete wall mass $m_{\mathrm{r,c}} = 2.376 \times 10^5\,\mathrm{kg}$ of temperature $T_{\mathrm{r,c}}$, while the concrete wall mass exchanges heat with the environment depending on the ambient temperature T_{amb}.

The main objective for the system control is to keep the room's air temperature $T_{\mathrm{r,a}}$ within a defined comfort range.

[8] In the following, we will treat mass flow rates produced by pumps as continuous controls with a minimum flow of 0.0 kg/s and not include the switching behavior into the problem formulation, as explained in the upcoming sections.

[9] For real applications, constant operation of fan coils would be inefficient and could instead be, e.g., set depending on pump operation, so that the fans are activated only when the pump is working, which would result in a similar system behavior but consume less electrical energy for fan operation.

Table 1
Specific heat capacities and densities of materials and media.

Substance	Specific heat capacity	Density
Water	$c_w = 4.12$ kJ/(kgK)	$\rho_w = 1.0 \times 10^3$ kg/m^3
Solar fluid (frost-proof)	$c_{sf} = 3.8$ kJ/(kgK)	$\rho_{sf} = 1.0 \times 10^3$ kg/m^3
Air	$c_a = 1.005$ kJ/(kgK)	$\rho_a = 1.2$ kg/m^3
Concrete	$c_c = 0.88$ kJ/(kgK)	$\rho_c = 2.2 \times 10^3$ kg/m^3

5.2. System modeling

For modeling of the system components, nonlinear gray-box models based on mass and energy balances are used that fulfill all necessary conditions regarding differentiability for use within derivative-based optimization methods, cf. [6]. If not stated explicitly, energy losses of the components to their surroundings are neglected. Material and media are assumed incompressible and with constant material properties as given in Table 1.

5.2.1. Solar collector model

To calculate the solar collector temperature T_{sc}, the single-node solar collector model

$$\begin{aligned}\dot{T}_{sc}(t) = \; & C_{sc}^{-1}[\dot{m}_{sc}(t)c_{sf}(T_{hx,hts}(t) - T_{sc}(t)) \\ & + \eta_{sc}A_{sc}I_g(t) - \alpha_{sc}A_{sc}(T_{sc}(t) - T_{amb}(t))]\end{aligned} \tag{7}$$

is used, where $C_{sc} = 2.6$ MJ/K is the lumped heat capacity of the solar collectors including the contained medium and $\alpha_{sc} = 1.4$ W/(m^{2}K) is the heat transfer coefficient for the heat losses of the collector to the environment, cf. [51].

5.2.2. Heat exchanger

Using a lumped model for the heat exchanger and a heat transfer coefficient $\alpha_{hx} = 3150.0$ W/(m^{2}K), the temperatures on the solar side $T_{hx,sc}$ and the water side $T_{hx,hts}$ of the heat exchanger can be calculated as

$$\begin{aligned}\dot{T}_{hx,sc}(t) &= \frac{\dot{m}_{sc}(t)c_{sf}(T_{sc}(t) - T_{hx,sc}(t)) - A_{hx}\alpha_{hx}(T_{hx,sc}(t) - T_{hx,hts}(t))}{m_{hx,sc}c_{sf}}, \\ \dot{T}_{hx,hts}(t) &= \frac{\dot{m}_{hx}(t)c_w(T_{hts,N}(t) - T_{hx,hts}(t)) + A_{hx}\alpha_{hx}(T_{hx,sc}(t) - T_{hx,hts}(t))}{m_{hx,hts}c_w},\end{aligned} \tag{8}$$

where $m_{hx,sc} = \rho_{sf}V_{hx} = 3.77$ kg and $m_{hx,hts} = \rho_w V_{hx} = 3.77$ kg.

5.2.3. HT water storage

To facilitate depiction of the temperature distribution in the stratified HT storage tank, we discretize the volume V_{hts} of the storage into $L = 4$ equally big layers so that the water mass of a layer is $m_{hts} = (\rho_w V_{hts})/L = 500$ kg and calculate an energy balance for each layer to determine its temperature, cf. [45], [16]. Magnitude and direction of mass flows between these layers depend on the current mass flow through the heat exchanger $\dot{m}_{hx}$ and ACM operation status $b_{acm} \in \{0, 1\}$, while we assume that $\dot{m}_{hx}(t) < \dot{m}_{acm,ht}$ at all times. The energy balance that determines the temperature $T_{hts,1}$ of the top layer can then be calculated according to (9), the temperatures for the the middle layers are determined by (10) and for the bottom layer by (11).

$$\begin{aligned}\dot{T}_{hts,1}(t) = \; & m_{hts}^{-1}[\dot{m}_{hx}(t)T_{hx,hts}(t) - (1 - b_{acm}(t))\dot{m}_{hx}(t)T_{hts,1}(t) \\ & + b_{acm}(t)((\dot{m}_{acm,ht} - \dot{m}_{hx}(t))T_{hts,2}(t) - \dot{m}_{acm,ht}T_{hts,1}(t))]\end{aligned} \tag{9}$$

$$\begin{aligned}\dot{T}_{hts,k}(t) = \; & m_{hts}^{-1}[(1 - b_{acm}(t))\dot{m}_{hx}(t)(T_{hts,k-1}(t) - T_{hts,k}(t)) \\ & + b_{acm}(t)(\dot{m}_{acm,ht} - \dot{m}_{hx}(t))(T_{hts,k+1}(t) - T_{hts,k}(t))], \quad k = 2, \ldots, L-1\end{aligned} \tag{10}$$

$$\begin{aligned}\dot{T}_{hts,L}(t) = \; & m_{hts}^{-1}[-\dot{m}_{hx}(t)T_{hts,L}(t) + (1 - b_{acm}(t))\dot{m}_{hx}(t)T_{hts,L-1}(t) \\ & + b_{acm}(t)(\dot{m}_{acm,ht}T_{acm,ht}(t) - (\dot{m}_{acm,ht} - \dot{m}_{hx}(t))T_{hts,L}(t))]\end{aligned} \tag{11}$$

5.2.4. Recooling tower

We assume the temperature of the water[10] returning from the sufficiently sized recooling tower to be of a constant difference $\Delta T_{rc} = 2$ K above the current ambient temperature T_{amb}. Depending on the current operation mode, the medium is either supported for recooling of the ACM or for direct cooling of the LT storage, as shown in the following sections.

5.2.5. Adsorption cooling machine

For operation of an ACM, upper and lower limits on a machine's inlet temperatures must be considered, while low Medium Temperature (MT) input temperatures and high HT and LT input temperatures are favorable conditions for efficient operation [13]. Within these limits, both the mean cooling power $\dot{Q}_{acm,lt}$ and the mean Coeffcient Of Performance (COP) of an ACM during one adsorption cycle are determined and can be depicted by, e.g., suitable (polynomial) curve fittings $f_{\dot{Q}_{acm,lt}}(\cdot)$ and $f_{cop}(\cdot)$. The energy balances for the relevant machine circuits can then be calculated as in

$$\begin{aligned}\dot{Q}_{acm,lt}(t) &= f_{\dot{Q}_{acm,lt}}(T_{hts,1}(t), T_{lts,1}(t), T_{amb}(t) + \Delta T_{rc}) \\ \dot{Q}_{acm,ht}(t) &= \frac{\dot{Q}_{acm,lt}(t)}{f_{cop}(T_{hts,1}(t), T_{lts,1}(t), T_{amb}(t) + \Delta T_{rc})}\end{aligned} \tag{12}$$

and using the mass flow rates $\dot{m}_{acm,lt} = 48$ kg/min and $\dot{m}_{acm,ht} = 41.8$ kg/min, the corresponding output temperatures of the circuits as in

$$\begin{aligned}T_{acm,lt}(t) &= T_{lts,1}(t) - \frac{\dot{Q}_{acm,lt}(t)}{c_w\dot{m}_{acm,lt}} \\ T_{acm,ht}(t) &= T_{hts,1}(t) - \frac{\dot{Q}_{acm,ht}(t)}{c_w\dot{m}_{acm,ht}}.\end{aligned} \tag{13}$$

The typical cyclic output temperature behavior of an ACM is not depicted by this model. However, the model becomes applicable due to the storages connected to the HT and LT side of the machine and the comparatively high mass flow rate of the MT circuit $\dot{m}_{acm,mt} = 85$ kg/min which dampens the effect of fluctuating output temperatures, see [43], [42] and [11].

5.2.6. LT water storage

The model for the LT storage is formulated analogously to the model of the HT storage. In addition to the current ACM status b_{acm} and water mass flow rate through the fan coils $\dot{m}_{fc,w}$, the mass flows in between the $M = 3$ modeled storage layers depend on the current status of the free cooling mode $b_{hd} \in \{0, 1\}$. The temperature $T_{lts,1}$ of the top layer of the LT storage is calculated as in (14), the temperature for the the middle layer is determined by (15) and for the bottom layer by (16).

$$\begin{aligned}\dot{T}_{lts,1}(t) = \; & m_{lts}^{-1}[\dot{m}_{fc,w}(t)T_{fc,w}(t) - (1 - b_{acm}(t) - b_{hd}(t))\dot{m}_{fc,w}(t)T_{lts,1}(t) \\ & + b_{acm}(t)((\dot{m}_{acm,lt} - \dot{m}_{fc,w}(t))T_{lts,2}(t) - \dot{m}_{acm,lt}T_{lts,1}(t)) \\ & + b_{hd}(t)((\dot{m}_{hd} - \dot{m}_{fc,w}(t))T_{hts,2}(t) - \dot{m}_{hd}T_{lts,1}(t))]\end{aligned} \tag{14}$$

$$\begin{aligned}\dot{T}_{lts,l}(t) = \; & m_{lts}^{-1}[(1 - b_{acm}(t) - b_{hd}(t))\dot{m}_{fc,w}(t)(T_{lts,l-1}(t) - T_{lts,l}(t)) \\ & + b_{acm}(t)(\dot{m}_{acm,lt} - \dot{m}_{fc,w}(t))(T_{lts,l+1}(t) - T_{lts,l}(t)) \\ & + b_{hd}(t)(\dot{m}_{hd} - \dot{m}_{fc,w}(t))(T_{hts,l+1}(t) - T_{hts,l}(t))], \quad l = 2, \ldots, M-1\end{aligned} \tag{15}$$

$$\begin{aligned}\dot{T}_{lts,M}(t) = \; & m_{lts}^{-1}[-\dot{m}_{fc,w}(t)T_{lts,M}(t) + (1 - b_{acm}(t) - b_{hd}(t))\dot{m}_{fc,w}(t)T_{lts,M-1}(t) \\ & + b_{acm}(t)(\dot{m}_{acm,lt}T_{acm,lt}(t) - (\dot{m}_{acm,lt} - \dot{m}_{fc,w}(t))T_{lts,M}(t)) \\ & + b_{hd}(t)(\dot{m}_{hd}T_{hd}(t) - (\dot{m}_{hd} - \dot{m}_{fc,w}(t))T_{lts,M}(t))]\end{aligned} \tag{16}$$

[10] Due to the simplicity of the recooler model, a system separation of the outdoor frost-proof recooling circuit from the indoor circuits is neglected here.

Table 2
Temperature boundaries for ACM operation.

Temperature	Lower bound	Upper bound
$T_{\mathrm{lts},1}$	$T_{\mathrm{lt,lb}} = 10\,^\circ\mathrm{C}$	$T_{\mathrm{lt,ub}} = 22\,^\circ\mathrm{C}$
$T_{\mathrm{hts},1}$	$T_{\mathrm{ht,lb}} = 55\,^\circ\mathrm{C}$	$T_{\mathrm{ht,ub}} = 95\,^\circ\mathrm{C}$
T_{amb}	$T_{\mathrm{amb,lb}} = 14\,^\circ\mathrm{C}$	$T_{\mathrm{amb,ub}} = 36\,^\circ\mathrm{C}$

5.2.7. Fan coil and room models

The fan coils are modeled as one single gas-liquid heat exchanger and by a single-node model with heat transfer coefficient $(A\alpha)_{\mathrm{fc}} = 475\,\mathrm{W/K}$. With $m_{\mathrm{fc,w}} = 3.6\,\mathrm{kg}$ the water mass and $m_{\mathrm{fc,a}} = 0.79\,\mathrm{kg}$ the air mass inside the fan coil, the temperatures of the water side $T_{\mathrm{fc,w}}$ and the air side $T_{\mathrm{fc,a}}$ of the fan coils are given by

$$\begin{aligned}\dot{T}_{\mathrm{fc,w}}(t) &= \frac{\dot{m}_{\mathrm{fc,w}}(t)c_{\mathrm{w}}(T_{\mathrm{lts},M}(t) - T_{\mathrm{fc,w}}(t)) + (A\alpha)_{\mathrm{fc}}(T_{\mathrm{fc,a}}(t) - T_{\mathrm{fc,w}}(t))}{m_{\mathrm{fc,w}}c_{\mathrm{w}}}\\ \dot{T}_{\mathrm{fc,a}}(t) &= \frac{\dot{m}_{\mathrm{fc,a}}(t)c_{\mathrm{a}}(T_{\mathrm{r,a}}(t) - T_{\mathrm{fc,a}}(t)) - (A\alpha)_{\mathrm{fc}}(T_{\mathrm{fc,a}}(t) - T_{\mathrm{fc,w}}(t))}{m_{\mathrm{fc,a}}c_{\mathrm{a}}}.\end{aligned} \tag{17}$$

The temperature of the room's air mass $T_{\mathrm{r,a}}$ and the temperature of the concrete mass of the wall $T_{\mathrm{r,c}}$ are determined by single-node models given in (18). The air and concrete masses exchange heat depending on the wall surface $A_{\mathrm{c}} = 540\,\mathrm{m}^2$ and a heat transfer coefficient $\alpha_{\mathrm{a,c}} = 10.0\,\mathrm{W/(m^2K)}$. The concrete mass additionally exchanges heat with the environment depending on the same factors and the ambient temperature T_{amb}.

$$\begin{aligned}\dot{T}_{\mathrm{r,a}}(t) &= \frac{\dot{m}_{\mathrm{fc,a}}(t)c_{\mathrm{a}}(T_{\mathrm{fc,a}}(t) - T_{\mathrm{r,a}}(t)) + A_{\mathrm{c}}\alpha_{\mathrm{a,c}}(T_{\mathrm{r,c}}(t) - T_{\mathrm{r,a}}(t)) + \dot{Q}_{\mathrm{load}}(t)}{m_{\mathrm{r,a}}c_{\mathrm{a}}}\\ \dot{T}_{\mathrm{r,c}}(t) &= \frac{A_{\mathrm{c}}\alpha_{\mathrm{a,c}}(T_{\mathrm{amb}}(t) - T_{\mathrm{r,c}}(t)) - A_{\mathrm{c}}\alpha_{\mathrm{a,c}}(T_{\mathrm{r,c}}(t) - T_{\mathrm{r,a}}(t))}{m_{\mathrm{r,c}}c_{\mathrm{c}}}\end{aligned} \tag{18}$$

5.2.8. Model summary

To summarize, the system is described by $n_x = 14$ differential states

$$x^\top = [T_{\mathrm{hts},\{1,2,3,4\}} \quad T_{\mathrm{hx},\{\mathrm{sc,hts}\}} \quad T_{\mathrm{lts},\{1,2,3\}} \quad T_{\mathrm{sc}} \quad T_{\mathrm{fc},\{\mathrm{w,a}\}} \quad T_{\mathrm{r},\{\mathrm{a,c}\}}], \tag{19}$$

which are all temperatures of system components, and influenced by $n_u = 3$ continuous controls, $n_b = 2$ binary controls and $n_c = 3$ time-varying parameters

$$\begin{aligned}u^\top &= [\dot{m}_{\mathrm{sc}} \quad \dot{m}_{\mathrm{hx}} \quad \dot{m}_{\mathrm{fc,w}}], \quad b^\top = [b_{\mathrm{acm}} \quad b_{\mathrm{hd}}],\\ c^\top &= [T_{\mathrm{amb}} \quad \dot{Q}_{\mathrm{load}} \quad I_{\mathrm{g}}].\end{aligned} \tag{20}$$

5.2.9. Operation conditions and constraints

System temperatures should not exceed $T_{\mathrm{ub}} = 105\,^\circ\mathrm{C}$ or go below $T_{\mathrm{lb}} = 5\,^\circ\mathrm{C}$. The limits on the inlet temperatures for operation of the ACM are given in Table 2. Since there is only one recooler present in the system, b_{acm} and b_{hd} cannot be active at the same time, i.e.,

$$b_{\mathrm{acm}}(t) + b_{\mathrm{hd}}(t) \leq 1. \tag{21}$$

Both ACM and HD are considered subject to dwell-time- and maximum-switching-constraints. The minimum operation time for the ACM should be 120 min and for the HD mode 60 min. The operation status of ACM and HD should not be changed more than $\sigma_{\mathrm{acm,max}} = \sigma_{\mathrm{hd,max}} = 4$ times within 24 h.

5.3. Optimal control problem formulation

Introducing a vector of $n_s = 21$ slack variables $s(t)^\top = [\Delta T_{\mathrm{r,a}}(t)\; s_{\mathrm{lb}}(t)^\top\; s_{\mathrm{ub}}(t)^\top s_T(t)^\top]$, the MIOCP for the STCS reads as

$$\underset{x(\cdot),u(\cdot),b(\cdot),s(\cdot)}{\mathrm{minimize}} \quad \int_{t_0}^{t_{\mathrm{f}}} u(t)^\top W_u u(t)\,\mathrm{d}t + \int_{t_0}^{t_{\mathrm{f}}} s(t)^\top W_s s(t)\,\mathrm{d}t \\ -w_{T_{\mathrm{hts},1}} T_{\mathrm{hts},1}(t_{\mathrm{f}}) + w_{T_{\mathrm{lts},3}} T_{\mathrm{lts},3}(t_{\mathrm{f}}) \tag{22a}$$

$$\mathrm{s.t.} \quad \textit{for } t \in [t_0, t_{\mathrm{f}}]: \\ (7)\text{-}(18), \tag{22b}$$

$$T_{\mathrm{r,a,lb}} \leq T_{\mathrm{r,a}}(t) + \Delta T_{\mathrm{r,a}}(t) \leq T_{\mathrm{r,a,ub}}, \tag{22c}$$

$$\epsilon \geq b_{\mathrm{acm}}(t)\left(\begin{pmatrix} T_{\mathrm{ht,lb}} - T_{\mathrm{hts},1}(t) \\ T_{\mathrm{lt,lb}} - T_{\mathrm{lts},1}(t) \\ T_{\mathrm{amb,lb}} - T_{\mathrm{amb}}(t)\end{pmatrix} - s_{\mathrm{lb}}(t)\right), \tag{22d}$$

$$\epsilon \geq b_{\mathrm{acm}}(t)\left(\begin{pmatrix} T_{\mathrm{hts},1}(t) - T_{\mathrm{ht,ub}} \\ T_{\mathrm{lts},1}(t) - T_{\mathrm{lt,ub}} \\ T_{\mathrm{amb}}(t) - T_{\mathrm{amb,ub}}\end{pmatrix} - s_{\mathrm{ub}}(t)\right), \tag{22e}$$

$$T_{\mathrm{lb}} \leq x(t) + s_T(t) \leq T_{\mathrm{ub}}, \tag{22f}$$

$$u_{\mathrm{lb}} \leq u(t) \leq u_{\mathrm{ub}}, \tag{22g}$$

$$b(t) \in \{0, 1\}^{n_{\mathrm{b}}}, \tag{22h}$$

$$\|b(t)\|_1 \leq 1, \tag{22i}$$

$$x(t_0) = x_0. \tag{22j}$$

The Lagrange term of the objective (22a) contains the sum of squares of the continuous controls u and the slack variables s, weighted by appropriate weighting matrices $W_u \in \mathbb{R}^{n_u \times n_u}$ and $W_s \in \mathbb{R}^{n_s \times n_s}$. While minimization of u favors energy efficient pump use, minimization of s reduces the deviation $\Delta T_{\mathrm{r,a}}$ of the room temperature $T_{\mathrm{r,a}}$ from an assumed comfort range $[T_{\mathrm{r,a,lb}}, T_{\mathrm{r,a,ub}}] = [20\,^\circ\mathrm{C}, 22\,^\circ\mathrm{C}]$ in (22c) as well as the use of s_{lb} and s_{ub} for relaxation of the temperature boundaries for ACM operation (22d) and (22e) and the use of s_{T} for relaxation of the general system temperature limits (22f). Constraint (22i) ensures that at most one binary control is active at a time.

The Mayer term contains the storage layer temperatures $T_{\mathrm{hts},1}(t_{\mathrm{f}})$ and $T_{\mathrm{lts},3}(t_{\mathrm{f}})$ at the final time point of the control horizon t_{f}, weighted by appropriate scalar weightings $w_{T_{\mathrm{hts},1}}$ and $w_{T_{\mathrm{lts},3}}$. Since high temperatures for $T_{\mathrm{hts},1}$ and low temperatures for $T_{\mathrm{lts},3}$ are generally advantageous, it is favorable to drive the system towards such a configuration.

The dynamics of the system that must be fulfilled are given in (22b). Path constraints that enforce compliance of the temperature bounds for ACM operation are introduced in (22d) and (22e) as smoothened vanishing constraints (cf. Section 2.4). While the boundaries on the continuous controls are given in (22g) as hard constraints, the boundaries on the system states in (22f) are formulated as soft constraints to preserve feasibility of the problem in case of state constraint violations, cf. [36]. In addition to that, the binary constraint (22h) must be fulfilled as well as the initial state constraint in (22j).

Additionally, the system is subject to dwell time constraints and maximum switching constraints, but which are not considered within solution of the NLPs but only within (CIA), since such constraints increase the size of the NLP very much, while they in fact shorten the solution time for (BnB) in *pycombina*.

5.4. Setup and implementation

The simulation study is implemented according to Section 4.1. Since we consider a solar driven system, we can assume more need for control interaction during day than during night, so for a control horizon of $t_{\mathrm{f}} = 24\,\mathrm{h}$ we set up the time grid using time steps of

$\Delta t_\mathrm{d} = 5$ min for day times in between 4:40 and 21:30[11], and longer time steps of $\Delta t_\mathrm{n} = 20$ min for night times in between 21:30 and 4:40 to reduce the number of optimization variables in the MINLP. The resulting time grid $\mathcal{G}_N$ consists of $N = 226$ time points. To avoid regeneration of the discretization in between MPC steps, we apply the time transformation as described in Section 2.3 so that the current time steps for each grid point enter the problem as time-varying parameters. In total, the MINLP contains 18000 continuous and 450 binary optimization variables as well as 12600 equality and 4950 inequality constraints.

The simulation model that we implement in Modelica can either be run using a PID- and set-point-based, conventional control scheme that is contained directly within the Modelica model, or it can be run using the control inputs from the solution of the MINLP. Further, the Modelica model includes a safety monitoring of the solar collectors temperature T_sc that assures that once T_sc exceeds a critical temperature, the mass flow rate through the collectors $\dot{m}_\mathrm{sc}$ is always set to its maximum value, e.g., when due to imprecise forecasts the temperature development inside the collectors is underestimated.

Mass flow rates requested by the controller that are below a pump's minimum flow rate introduced in Section 5.1 are treated by pulsing the corresponding mass transported over an interval for the requested flow rate within two shorter pulses at the corresponding minimum flow rate. The necessary pulsing time $\Delta t_{k,\mathrm{p}}$ within an interval k is then obtained from the simple relation

$$\Delta t_{k,\mathrm{p}} = \frac{\Delta t_k \dot{m}_k}{2 \cdot \dot{m}_\mathrm{min}} \tag{23}$$

while the pulsing of mass is initiated at time points t_k and $t_k + \frac{\Delta t_k}{2}$. This approach becomes applicable only due to the short start-up-time of the pumps and is not suitable for, e.g., operation of the ACM.

5.5. Conduction of the simulation study

Using the implementations and methods described in the previous sections, the simulation study for the system is conducted as follows:

1 Starting from an initial state of the system at 2:00 am at night of

$$x_0^\top = [55.0 \;\; 51.7 \;\; 48.3 \;\; 45.0 \;\; 20.0 \;\; 19.0 \;\; 18.0 \;\; 20.0 \;\; 21.0 \;\; 21.0 \;\; 22.0 \;\; 22.0 \;\; 20.0 \;\; 20.0]\,^\circ\mathrm{C}, \tag{24}$$

an initial guess for the MINLP solution algorithm is obtained from simulation in OpenModelica using the included conventional controller and forecast values for solar irradiation $I_\mathrm{g,fc}$, ambient temperature $T_\mathrm{amb,fc}$ and heat load $\dot{Q}_\mathrm{load,fc}$.
2 Then, the MINLP is solved using the initial guess obtained in the previous step and the forecasts $I_\mathrm{g,fc}$, $T_\mathrm{amb,fc}$ and $\dot{Q}_\mathrm{load,fc}$.
3 After that, a simulation is conducted in OpenModelica applying the control inputs obtained from the solution of the MINLP, but using measured values for solar irradiation $I_\mathrm{g,m}$, ambient temperature $T_\mathrm{amb,m}$ and heat load $\dot{Q}_\mathrm{load,m}$ that deviate from the forecast values at a varying extend.
4 The result of the simulation for the upcoming time point is then used as initial state vector x_1 for the next MPC steps and corresponding MINLP.[12]

We repeat this process to simulate an MPC loop of 1000 steps, which corresponds to more than 100 h of system operation time. The profile of forecasted and corresponding measured solar irradiation I_g, ambient temperature T_amb and heat load $\dot{Q}_\mathrm{load}$ used within this study are depicted in Fig. 5.[13] The heat load is calculated for both the measured load $\dot{Q}_\mathrm{load,m}$ and forecasted load $\dot{Q}_\mathrm{load,fc}$ by a simple relation to a reference temperature of $T_\mathrm{ref} = 22\,^\circ$C as in

$$\dot{Q}_{\mathrm{load},\{\mathrm{fc,m}\}}(t) = \begin{cases} 0.3\,\mathrm{kW}, & \text{if} T_{\mathrm{amb},\{\mathrm{fc,m}\}}(t) \leq T_\mathrm{ref}, \\ 0.3\,\mathrm{kW} + 1.2\,\dfrac{\mathrm{kW}}{\mathrm{K}}(T_{\mathrm{amb},\{\mathrm{fc,m}\}}(t) - T_\mathrm{ref}), & \text{if } \; T_{\mathrm{amb},\{\mathrm{fc,m}\}}(t) > T_\mathrm{ref}. \end{cases} \tag{25}$$

5.6. Simulation results

The result of the simulation is shown in Fig. 6. The upper two plots show the development of the HT and LT side of the system on selected system states. The second lowest plot shows the continuous controls, while the bottom plot depicts binary controls.

The graph shows that the MPC is able to drive the system in a desirable way. The controller heats up the solar collector temperature T_sc at low pump operation $\dot{m}_\mathrm{sc}$ and $\dot{m}_\mathrm{hx}$ and increases pump operation with rising temperature T_sc to charge the HT storage, observable from increasing storage temperatures $T_\mathrm{hts,1}$ and $T_\mathrm{hts,4}$. The ACM is operated to decrease the temperature of the LT storage depicted by the storage layer temperatures $T_\mathrm{lts,1}$ and $T_\mathrm{lts,3}$. For doing this, the controller operates the ACM ($b_\mathrm{acm} = 1$) not only during occurrence of thermal loads, but also at different points in the early day or during night to benefit from the low ambient temperature T_amb and resulting in a high COP for the ACM to predictively regenerate the LT storage in preparation for high loads of an upcoming day, cf. [11]. Free cooling ($b_\mathrm{hd} = 1$) is thoroughly used during times of low ambient temperature to cool down room air and concrete mass especially during nights. Nevertheless, the system is not completely able to keep the room temperature $T_\mathrm{r,a}$ within its desired comfort region at all times, which is due to the (intentionally chosen) comparatively high heat load affecting the system compared to the system's cooling capacities.

Despite mismatches between forecasted and occurring solar irradiation, ambient temperature and heat load profiles, the results in Fig. 6 show that the MPC is capable of controlling the system sufficiently. However, imprecise forecasts can cause suboptimal system operation, as shown, e.g., by the fast switching of the ACM around hour 80. Also, underestimation of the solar irradiation might lead to an unexpected increase in the solar collector temperature, and with this, to an intervention of the solar temperature safety monitoring or violation of temperature boundaries.

5.7. Constraint satisfaction

Fig. 6 shows that the dwell time constraints are always fulfilled. Regarding constraint satisfaction, violations of the operation tem-

[11] These time points given in Central European Summer Time (CEST) correspond approximately to the earliest possible sunrise and latest possible sunset throughout the year in Southern Germany, where the original STCS is located.

[12] Within this study, we assume that all states can be perfectly measured and neglect explicit treatment of model-plant-mismatch for simplicity. Within real-world applications however, suitable state estimation methods should be applied.

[13] For the sake of simplicity, we omit measurement-based updates of forecasted values within this study. However, such considerations should be made within real-world applications for improving a controller's performance.

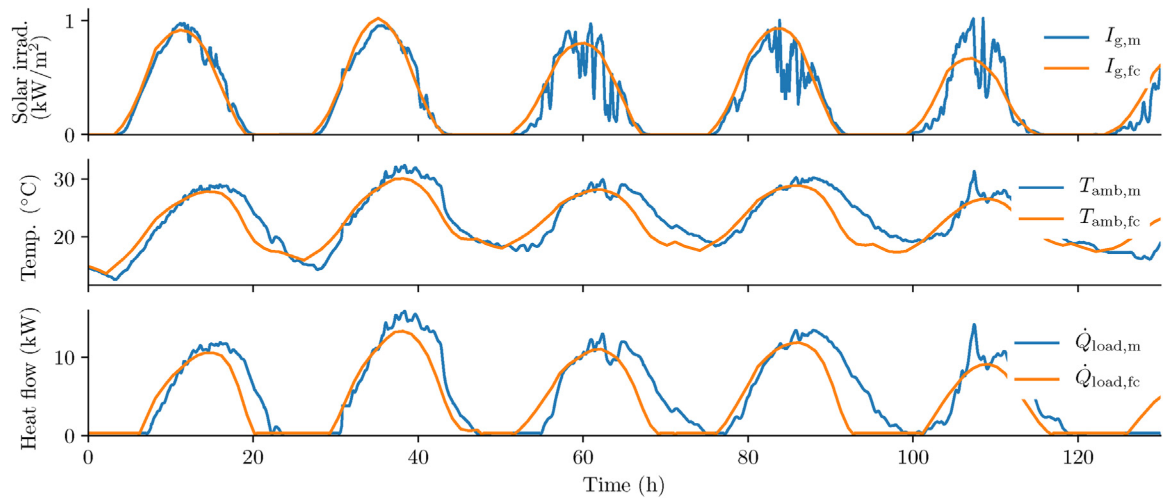

Fig. 5. Profiles of forecasted (fc) and corresponding measured (m) solar irradiation I_g, ambient temperature T_{amb} and heat load $\dot{Q}_{load}$ considered within this study.

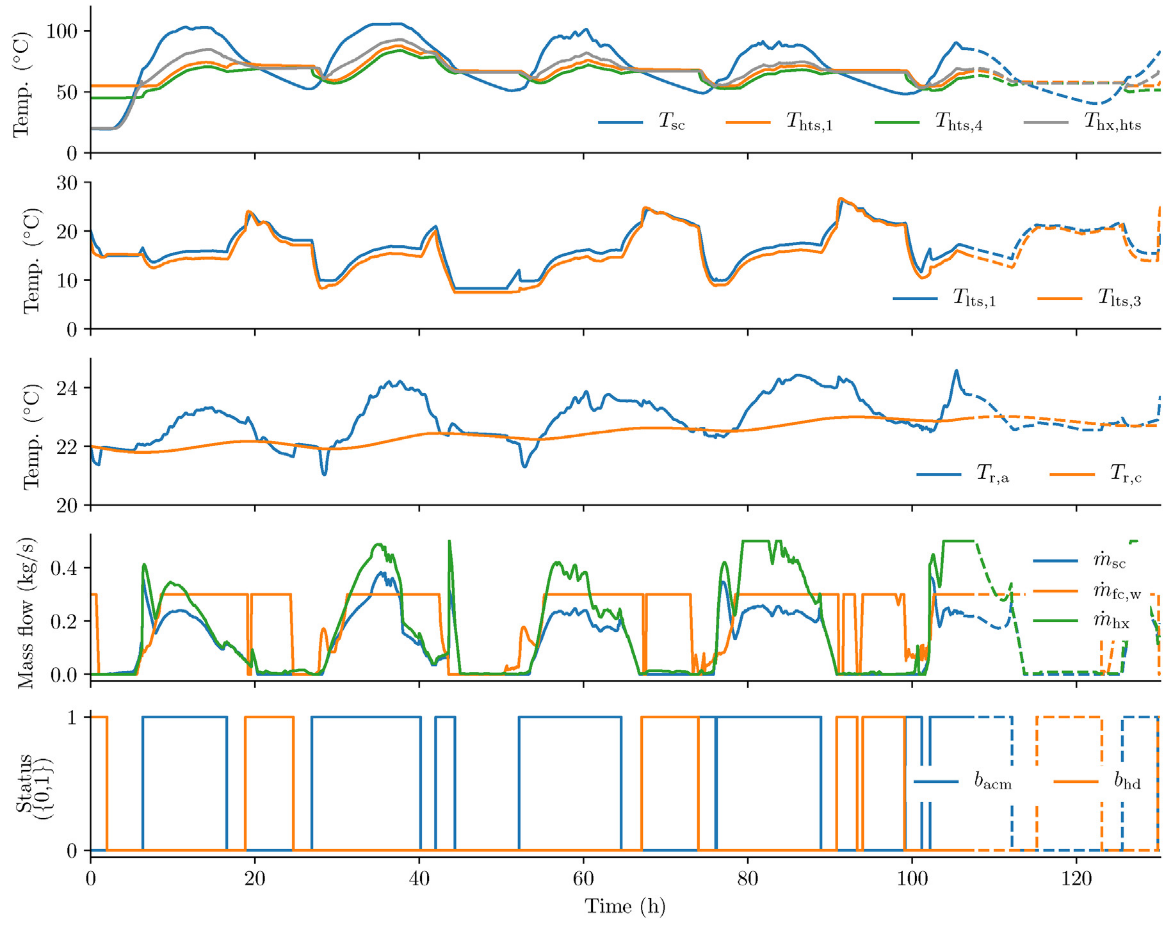

Fig. 6. Results of the MPC simulation loop for storage and room temperatures, pump operation and ACM and HD activation. The solid lines show the results of the MPC simulation loop, the dashed lines indicate the results of the MIOCP at the last simulation step.

perature constraints $T_{ht,lb}$ and $T_{lt,lb}$ as in

$$\Delta T_{\{ht,lt\},lb}(t) = \begin{cases} \min(0, T_{\{hts,lts\},1}(t) - T_{\{ht,lt\},lb}), & \text{if } b_{acm}(t) = 1, \\ 0, & \text{else,} \end{cases} \tag{26}$$

as well as a violation of the maximum system temperature T_{ub} for the solar collector temperature T_{sc} result in utilization of the corresponding slack variables for relaxation of the constraints to preserve feasibility of the NLPs. Apart from these, no operations constraints are violated. The actual values of violations are shown in Fig. 7.

One reason for constraint violation can be found within the approximation of the binary controls within CIA, which might result in scheduling of machine operation in times where corresponding operation conditions are not (yet) completely met, cf.

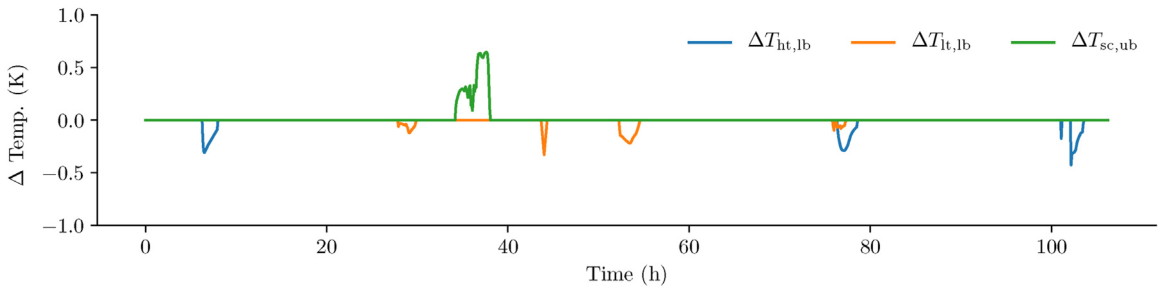

Fig. 7. Violations of ACM operation constraints and system temperature boundaries during MPC.

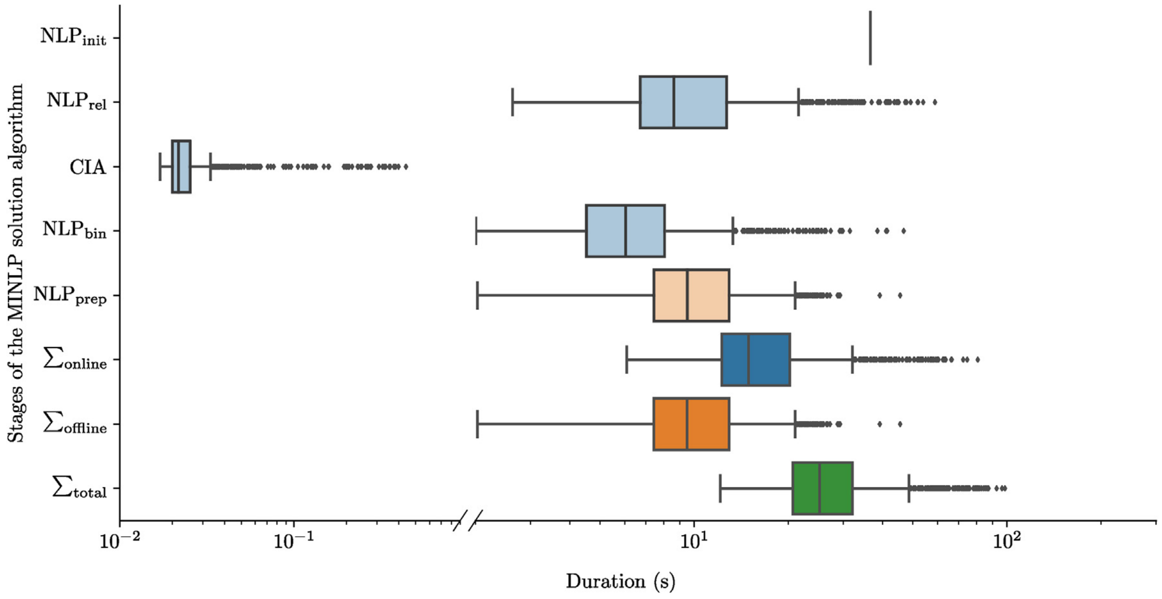

Fig. 8. Box-plot showing the runtimes of the several stages of the MINLP solution process.

[12]. Also, mismatches between forecasted and measured time-varying parameter values result in under- and/or overestimation of system temperatures. In order to avoid such violations, larger safety margins for the constraints within the OCP formulation could be introduced, or the quality of the utilized forecasts could be improved.

In case of larger constraint violations, internal safety measures similar to the solar temperature monitoring included in this study can be utilized to perform, e.g., a safety shutdown of a machine. For the components used within this study, however, we can presume constraint violations of the presented dimension, which are always smaller than 1 K, acceptable.

5.8. *Runtime of solution stages*

Fig. 8 shows a box-plot for the individual runtimes for the several solution stages of the algorithm as well as the total duration of the online and offline parts and the overall solution time of one MINLP solution process. As customary, the box ranges from the 25% to the 75% quartile. The whiskers of the boxes extend these by 1.5 times the distance between the 75% and 25% quartile, which is the so-called Inter-Quartile Range (IQR). The remaining values outside of this range are regarded as outliers, denoted as dots within the plot. The median is denoted as a vertical line, however for $\mathrm{NLP_{init}}$ which is evaluated only once at the start of the MPC loop the vertical line indicates the duration of this single run. Fig. 9 shows an additional plot for the relation between online and total solution times. The corresponding minimum and maximum values shown in the plots as well as the values of the median and quartiles are depicted in Table 3.

From the plots and table given, it can be observed that in 50% of the cases solving the MINLP takes less than 26 s, and less than 33 s for more than 75%. In 50% of the cases, the online part of the algorithm takes up less than 63% of the total runtime, and less than 69% in 75% of the cases. Both the maxima of the online and total runtimes are far below the smallest element of the discretization time grid $\mathcal{G}_N$.

All CIA solution times obtained using the branch-and-bound method of *pycombina* after preprocessing are below 0.5 s. The majority of the algorithm runtime is spent within solution of the NLP problems, while similar shares are spent within preparation of the next NLP solution step $\mathrm{NLP_{prep}}$ and the solution of $\mathrm{NLP_{rel}}$. All NLPs solved fully within this study converged to a feasible optimal solution.

Since it has been shown in [12] that the solution time for a similar but less complex problem can already increase up to several hours when a general MINLP solver is used, we apply Bonmin with default settings for solution of the first MINLP of the simulation study for comparison, however neglecting the complex dwell-time- and maximum-switching-constraints for simplicity. Still, it takes approx. 10 hrs for Bonmin to find an optimal integer solution for the problem, while our decomposition approach is able to find a solution of comparable quality in approx. 50 s.

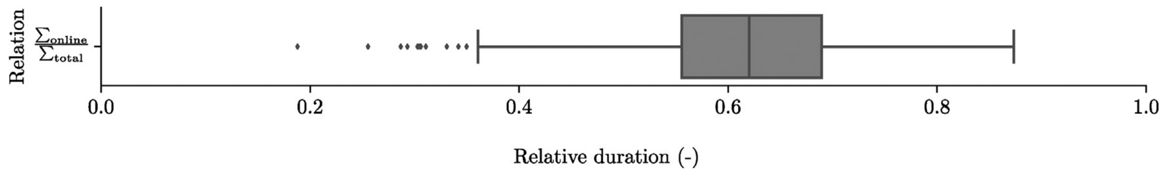

Fig. 9. Durations of the online parts of the algorithm relative to the total solution times.

Table 3
Numerical values of the properties shown in Fig. 8.

	$\sum_{\text{offline}}$ (s)	$\sum_{\text{online}}$ (s)	$\sum_{\text{total}}$ (s)	CIA (s)	NLP_{bin} (s)	NLP_{init} (s)	NLP_{prep} (s)	NLP_{rel} (s)
Minimum	2.027	6.093	12.139	0.017	2.009	-	2.027	2.631
$Q_{0.25}$	7.426	12.254	20.695	0.020	4.518	-	7.426	6.710
Median	9.498	14.915	25.198	0.022	6.036	-	9.498	8.605
$Q_{0.75}$	12.927	20.228	32.044	0.025	8.038	-	12.927	12.680
Maximum	45.556	80.662	98.520	0.440	46.767	36.569	45.556	58.872

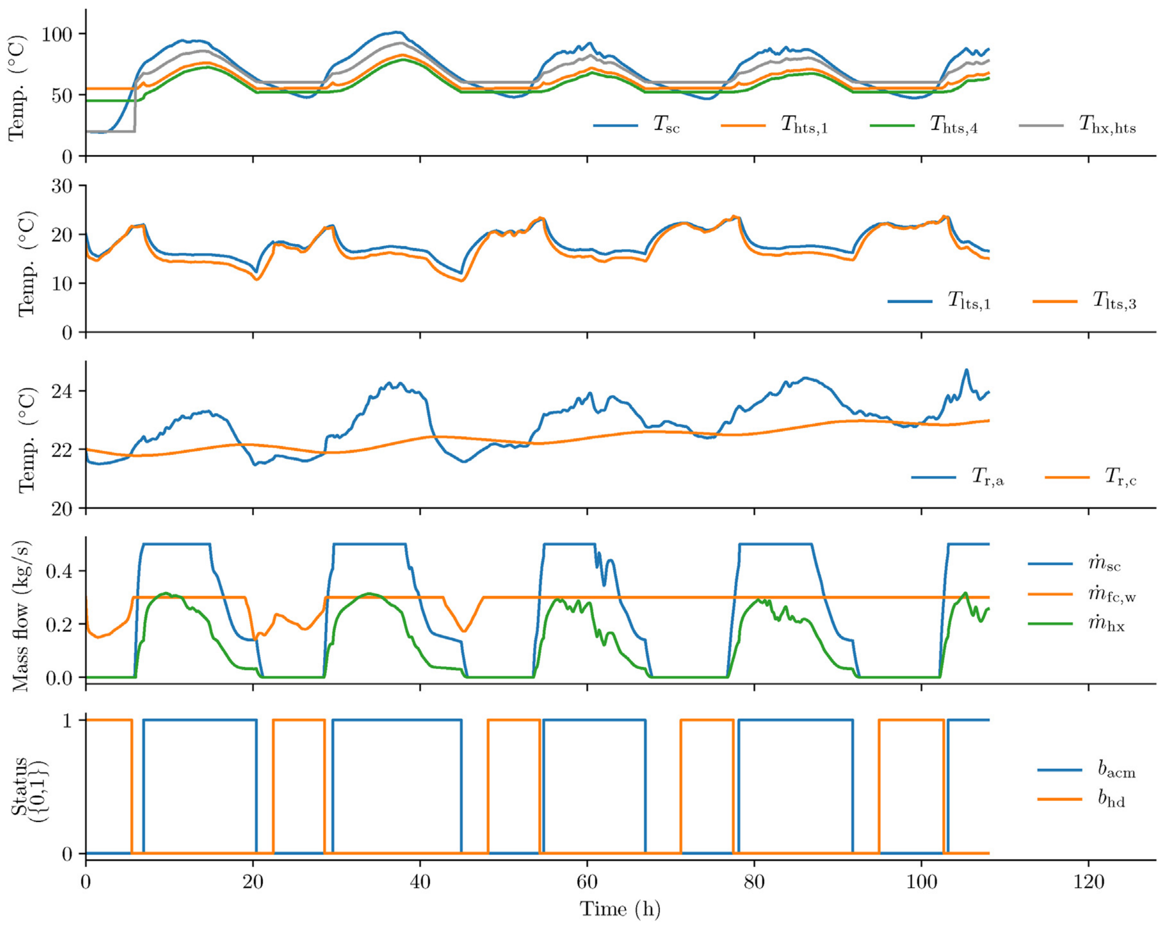

Fig. 10. Results of a simulation for the conventional control scheme for storage and room temperatures, pump operation and ACM and HD activation.

Amongst others, such solution times for a problem of the given complexity renders the algorithm suitable for applications within process control and control of energy and HVAC systems. This field of applications could possibly be enlarged if the runtime of the algorithm was further reduced, e.g., by application of tailored NLP solution methods. Though Ipopt is tailored for solution of large-scale, sparse systems [48], further possible reductions in runtime could be achieved by application of suitable Sequential Quadratic Programming (SQP) solvers which are intrinsically more suitable for warm-starting, cf. [36]. In addition to that, the C-code generation and parallel computation facilities within CasADi could be utilized to further speed up computations.

5.9. *Comparison to a conventional control scheme*

Finally, a system simulation using the Conventional Controller (CC) as applied for initial guess generation of the first MPC step is conducted in OpenModelica and its performance and behavior is compared to the simulated MPC loop presented in the previous sections.

Since the CC is tuned for scenarios of high heat load occurrence as used within this study, behavior and performance of the CC shown in Fig. 10 at first glance appear quite similar to the results of the MPC. However, a closer investigation reveals several aspects that illustrate the benefits of the MPC application, as described in the following.

5.9.1. *Solar collector temperature*

Since the CC has no information on the future development of the temperature T_{sc}, the controller needs to set a high mass flow through the solar collectors in order to prevent possible overheating caused by fast temperature increases, as shown on the high values of the control $\dot{m}_{sc}$.

The MPC on the other hand can directly take into account the estimated development of T_{sc}, and therefore typically increases T_{sc} earlier during the day and to a comparatively higher temperature. This can be observed, e.g., during the second and third day of the simulation study. Due to the quadratic contributions of $\dot{m}_{sc}$ to the MPC objective which approximates the superlinear increase of electricity consumption of pumps at increasing speed, collector operation is realized at comparatively lower mass flows and therefore in a more energy-efficient way.

5.9.2. *Rules for ACM and HD activation*

Within the CC, specific and hierarchical rules for activation of ACM and HD cooling based on current system temperatures need to be defined. As observable in Fig. 10, this results in utilization of HD cooling during night and ACM cooling during day.

In contrast to that, the MPC can make flexible decisions based on the current situation, which can be observed, e.g., during the night between the second and third day of the study. There, since the comfort region of $T_{r,a}$ is already reached, the MPC favors ACM cooling for preparation of the LT storage for upcoming loads during efficient ACM operation conditions, contrary to the CC which utilizes HD cooling for further reducing the room temperature.

5.9.3. *Reference temperature and utilization of the comfort range*

For the CC, the comfort range for the room temperature allows for setting of a reference temperature and a corresponding hystereses at which cooling needs to be activated or deactivated, respectively. Once the cooling is active, the mass flow $\dot{m}_{fc,w}$ is determined based on the current deviation of $T_{r,a}$ from its reference value. However, this behavior is fixed and independent of the expected heat loads of the upcoming hours or days.

Within the MPC, the comfort range introduces comparatively more flexibility since the controller can decide on its utilization in a situation-dependent way. This allows to decrease room temperature, e.g., in time of low heat loads and in this way prepare for expected higher loads during the upcoming day. Here, setting a lower reference temperature for the CC however could result in a room temperature that is generally lower than necessary, resulting in a unnecessary high energy usage.

6. Conclusions and outlook

In this work, we presented an algorithm for MPC of switched nonlinear systems under combinatorial constraints and a corresponding new toolbox for solving CIA problems with special focus on application within MPC including preprocessing heuristics for complexity reduction of CIA problems. Within a simulated MPC loop, the implemented algorithm was applied for the case study of an STCS with nonlinear system behavior and combinatorial constraints whose operation conditions are subject to uncertainty. It has been shown that the algorithm is capable to efficiently control the system while preserving sufficient constraint satisfaction and algorithm runtimes.

Future work should focus on further runtime reduction by applying more tailored NLP solution methods. Also, the application of the algorithm for other systems or MPC applications with possibly increasing complexity may be investigated and finally the physical installation of the STCS presented within this work is desirable.

Conflict of interest

None.

Acknowledgments

This research was supported by the European Research Council (ERC) via H2020-MODEST (647 573), by the German Federal Ministry for Economic Affairs and Energy (BMWi) via eco4wind (0324125B) and DyConPV (0324166B), by DFG via Research Unit FOR 2401, GRK 2297 and SPP 1962, and by the State Ministry of Baden-Wuerttemberg for Sciences, Research and Arts (Az: 22-7533.-30-20/9/3), which is gratefully acknowledged.

References

[1] P.R. Amestoy, I.S. Duff, J. Koster, J.Y. L'Excellent, A fully asynchronous multifrontal solver using distributed dynamic scheduling, SIAM J. Matrix Anal. Appl. 23 (2001) 15–41, http://dx.doi.org/10.1137/S0895479899358194.
[2] P.R. Amestoy, A. Guermouche, J.Y. L'Excellent, S. Pralet, Hybrid scheduling for the parallel solution of linear systems, Parallel Comput. 32 (2006) 136–156, http://dx.doi.org/10.1016/j.parco.2005.07.004.
[3] J.A.E. Andersson, J. Gillis, G. Horn, J.B. Rawlings, M. Diehl, CasADi: a software framework for nonlinear optimization and optimal control, Math. Program. Comput. (2018), http://dx.doi.org/10.1007/s12532-018-0139-4.
[4] U. Bau, A.L. Braatz, F. Lanzerath, M. Herty, A. Bardow, Control of adsorption chillers by a gradient descent method for optimal cycle time allocation, Int. J. Refrigeration 56 (2015) 52–64, http://dx.doi.org/10.1016/j.ijrefrig.2015.03.026.
[5] A. Bemporad, M. Morari, Control of systems integrating logic, dynamics, and constraints, Automatica 35 (1999) 407–427, http://dx.doi.org/10.1016/S0005-1098(98)00178-2.
[6] L. Biegler, Nonlinear Programming, Society for Industrial and Applied Mathematics, 2010, http://dx.doi.org/10.1137/1.9780898719383.
[7] T. Binder, L. Blank, H.G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, O. von Stryk, Introduction to model based optimization of chemical processes on moving horizons, in: M. Grötschel, S.O. Krumke, J. Rambau (Eds.), Online Optimization of Large Scale Systems, Springer Berlin Heidelberg, 2001, pp. 295–339, http://dx.doi.org/10.1007/978-3-662-04331-8_18.
[8] H. Bock, K. Plitt, A multiple shooting algorithm for direct solution of optimal control problems, IFAC Proc. 17 (1984) 1603–1608, http://dx.doi.org/10.1016/S1474-6670(17)61205-9.
[9] H.G. Bock, C. Kirches, A. Meyer, A. Potschka, Numerical solution of optimal control problems with explicit and implicit switches, Optimiz. Methods Softw. 33 (2018) 450–474, http://dx.doi.org/10.1080/10556788.2018.1449843.
[10] P. Bonami, L.T. Biegler, A.R. Conn, G. Cornuéjols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, A. Wächter, An algorithmic framework for convex mixed integer nonlinear programs, Discrete Optimiz. 5 (2008) 186–204.
[11] A. Bürger, P. Sawant, M. Bohlayer, A. Altmann-Dieses, M. Braun, M. Diehl, Efficient operation scheduling for adsorption chillers using predictive optimization-based control methods, IOP Conference Series: Materials Science and Engineering 257 (2017) 012007, http://dx.doi.org/10.1088/1757-899X/257/1/012007.
[12] A. Bürger, C. Zeile, A. Altmann-Dieses, S. Sager, M. Diehl, An algorithm for mixed-integer optimal control of solar thermal climate systems with MPC-capable runtime, 2018 European Control Conference (ECC) (2018) 1379–1385, http://dx.doi.org/10.23919/ECC.2018.8550424.
[13] W.S. Chang, C.C. Wang, C.C. Shieh, Experimental study of a solid adsorption cooling system using flat-tube heat exchangers as adsorption bed, Appl. Therm. Eng. 27 (2007) 2195–2199, http://dx.doi.org/10.1016/j.applthermaleng.2005.07.022.
[14] L.S. Dias, R.C. Pattison, C. Tsay, M. Baldea, M.G. Ierapetritou, A simulation-based optimization framework for integrating scheduling and model predictive control, and its application to air separation units, Comput. Chem. Eng. 113 (2018) 139–151, http://dx.doi.org/10.1016/j.compchemeng.2018.03.009.
[15] T. Ebrahim, S. Subramaianan, S. Engell, Hybrid NMPC for switching systems applied to a supermarket refrigeration system, 2018 European Control Conference (ECC) (2018), http://dx.doi.org/10.23919/ECC.2018.8550609.
[16] U. Eicker, Solar technologies for buildings, John Wiley & Sons, Chichester, 2003, http://dx.doi.org/10.1002/0470868341.
[17] F. Farshidian, M. Kamgarpour, D. Pardo, J. Buchli, Sequential linear quadratic optimal control for nonlinear switched systems, IFAC-PapersOnLine 50 (2017) 1463–1469, http://dx.doi.org/10.1016/j.ifacol.2017.08.291, 20th IFAC World Congress.

[18] D. Frick, A. Domahidi, M. Morari, Embedded optimization for mixed logical dynamical systems, Comput. Chem. Eng. 72 (2015) 21–33, http://dx.doi.org/10.1016/j.compchemeng.2014.06.005, a Tribute to Ignacio E. Grossmann.
[19] M. Gerdts, A variable time transformation method for mixed-integer optimal control problems, Optimal Control Appl. Methods 27 (2006) 169–182, http://dx.doi.org/10.1002/oca.778.
[20] A. Gleixner, L. Eifler, T. Gally, G. Gamrath, P. Gemander, R.L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. Miltenberger, B. Müller, M.E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, F. Serrano, Y. Shinano, J.M. Viernickel, S. Vigerske, D. Weninger, J.T. Witt, J. Witzig, The SCIP Optimization Suite 5.0. Technical Report 17-61. ZIB. Takustr.7, 14195 Berlin, 2017, Last accessed June 28, 2018 https://opus4.kobv.de/opus4-zib/files/6629/scipopt-50.pdf.
[21] Gurobi Optimization Inc, Gurobi optimizer reference manual, 2016, Last accessed June 28, 2018 http://www.gurobi.com.
[22] M. Heidarinejad, J. Liu, P.D. Christofides, Economic model predictive control of switched nonlinear systems, Syst. Contr. Lett. 62 (2013) 77–84, http://dx.doi.org/10.1016/j.sysconle.2012.11.002.
[23] HSL, 2017. A collection of Fortran codes for large scale scientific computation. http://www.hsl.rl.ac.uk/. Last accessed June 28, 2018.
[24] IBM Corp., 2018. Cplex optimizer. https://www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer. Last accessed June 28, 2018.
[25] Jung, M., 2013. Relaxations and Approximations for Mixed-Integer Optimal Control. Ph.D. dissertation. Interdisciplinary Center for Scientific Computing, Heidelberg University. http://doi.org/10.11588/heidok.00016036.
[26] M. Jung, C. Kirches, S. Sager, On Perspective Functions and Vanishing Constraints in Mixed-Integer Nonlinear Optimal Control, in: M. Jünger, G. Reinelt (Eds.), Facets of Combinatorial Optimization, Springer Berlin Heidelberg, 2013, pp. 387–417, http://dx.doi.org/10.1007/978-3-642-38189-8_16.
[27] A. Khakimova, A. Kusatayeva, A. Shamshimova, D. Sharipova, A. Bemporad, Y. Familiant, A. Shintemirov, V. Ten, M. Rubagotti, Optimal energy management of a small-size building via hybrid model predictive control, Energy Build. 140 (2017) 1–8, http://dx.doi.org/10.1016/j.enbuild.2017.01.045.
[28] C. Kirches, Fast Numerical Methods for Mixed-Integer Nonlinear Model-Predictive Control, Vieweg+Teubner Verlag, Wiesbaden, 2011, http://dx.doi.org/10.1007/978-3-8348-8202-8.
[29] K. Kobayashi, J. Imura, Deterministic finite automata representation for model predictive control of hybrid systems, J. Process Control 22 (2012) 1670–1680, http://dx.doi.org/10.1016/j.jprocont.2012.07.003.
[30] J. Kreiss, L. Bako, E. Blanco, Optimal control of discrete-time switched linear systems via continuous parameterization, IFAC-PapersOnLine 50 (2017) 15331–15336, http://dx.doi.org/10.1016/j.ifacol.2017.08.2453, 20th IFAC World Congress.
[31] B. Lie, S. Bajracharya, A. Mengist, L. Buffoni, A. Palanisamy, M. Sjölund, A. Asghar, A. Pop, P. Fritzson, API for accessing OpenModelica models from Python, Proceedings of 9th EUROSIM Congress on Modelling and Simulation (2016).
[32] E. Luchini, A. Schirrer, M. Kozek, A hierarchical MPC for multi-objective mixed-integer optimisation applied to redundant refrigeration circuits, IFAC-PapersOnLine 50 (2017) 9058–9064, http://dx.doi.org/10.1016/j.ifacol.2017.08.1629, 20th IFAC World Congress.
[33] N.A. Manaf, A. Qadir, A. Abbas, The hybrid MPC-MINLP algorithm for optimal operation of coal-fired power plants with solvent based post-combustion CO_2 capture, Petroleum 3 (2017) 155–166, http://dx.doi.org/10.1016/j.petlm.2016.11.009, Carbon Capture and Storage (CCUS).
[34] B. Mayer, M. Killian, M. Kozek, A branch and bound approach for building cooling supply control with hybrid model predictive control, Energy and Buildings 128 (2016) 553–566, http://dx.doi.org/10.1016/j.enbuild.2016.07.027.
[35] P.R. Mendes, J.M. Maestre, C. Bordons, J.E. Normey-Rico, A practical approach for hybrid distributed MPC, J. Process Control 55 (2017) 30–41, http://dx.doi.org/10.1016/j.jprocont.2017.01.001.
[36] J.B. Rawlings, D.Q. Mayne, M.M. Diehl, Model Predictive Control: Theory, Computation, and Design, 2nd edition, Nob Hill, 2017.
[37] J.B. Rawlings, N.R. Patel, M.J. Risbeck, C.T. Maravelias, M.J. Wenzel, R.D. Turney, Economic MPC and real-time decision making with application to large-scale HVAC energy systems, Comput. Chem. Eng. 114 (2018) 89–98, http://dx.doi.org/10.1016/j.compchemeng.2017.10.038, fOCAPO/CP, 2017.
[38] J.B. Rawlings, M.J. Risbeck, Model predictive control with discrete actuators: Theory and application, Automatica 78 (2017) 258–265, http://dx.doi.org/10.1016/j.automatica.2016.12.024.
[39] S. Sager, Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control, J. Process Control 19 (2009) 1238–1247, http://dx.doi.org/10.1016/j.jprocont.2009.03.008, special Section on Hybrid Systems: Modeling, Simulation and Optimization.
[40] S. Sager, A benchmark library of mixed-integer optimal control problems, in: J. Lee, S. Leyffer (Eds.), Mixed Integer Nonlinear Programming, Springer New York, 2012, pp. 631–670, http://dx.doi.org/10.1007/978-1-4614-1927-3_22.
[41] S. Sager, M. Jung, C. Kirches, Combinatorial Integral Approximation, Math. Methods Oper. Res. 73 (2011) 363–380, http://dx.doi.org/10.1007/s00186-011-0355-4.
[42] P. Sawant, D. Doan, Modelling and simulation of microscale trigeneration systems based on real-life experimental data, IFAC-PapersOnLine 50 (2017) 3238–3243, http://dx.doi.org/10.1016/j.ifacol.2017.08.452, 20th IFAC World Congress.
[43] P.A. Sawant, J. Pfafferott, Experimental investigation of a real-life microscale trigeneration system using adsorption cooling, reversible heat-pump and a cogeneration unit, 7th International Conference on Experiments/Process/System Modeling/Simulation/Optimization IC-EPSMSO (2017).
[44] G. Schweiger, P.O. Larsson, F. Magnusson, P. Lauenburg, S. Velut, District heating and cooling systems - framework for Modelica-based simulation and dynamic optimization, Energy 137 (2017) 566–578, http://dx.doi.org/10.1016/j.energy.2017.05.115.
[45] G. Streckiene, V. Martinaitis, P. Vaitiekunas, Simulation of thermal stratification in the heat storage for CHP plant, 8th International Conference on Environmental Engineering (2011).
[46] The Open Source Modelica Consortium, OpenModelica, 2017, Last accessed June 28, 2018 https://www.openmodelica.org.
[47] T.H. Tsang, D.M. Himmelblau, T.F. Edgar, Optimal control via collocation and non-linear programming, Int. J. Control 21 (1975) 763–768, http://dx.doi.org/10.1080/00207177508922030.
[48] S. Vigerske, A. Wächter, C. Laird, Y. Kawajir, Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT, 2016, Last accessed June 28, 2018 https://projects.coin-or.org/Ipopt/browser/stable/3.12/Ipopt/doc/documentation.pdf.
[49] A. Wächter, L.T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Math. Progam. 106 (2006) 25–57, http://dx.doi.org/10.1007/s10107-004-0559-y.
[50] J. Wenzel, J. Rhinelander, D. Moldovan, pybind11 - seamless operability between C++11 and Python, 2016, Last accessed June 28, 2018 https://github.com/pybind/pybind11.
[51] V. Wesselak, T. Schabbach, T. Link, J. Fischer, Regenerative Energietechnik, 2 ed., Springer Vieweg, Berlin, Heidelberg, 2013, http://dx.doi.org/10.1007/978-3-642-24165-9.
[52] A. Zanelli, R. Quirynen, J. Jerez, M. Diehl, A homotopy-based nonlinear interior-point method for NMPC, IFAC-PapersOnLine 50 (2017) 13188–13193, http://dx.doi.org/10.1016/j.ifacol.2017.08.2175, 20th IFAC World Congress.
[53] C. Zeile, T. Weber, S. Sager, Combinatorial integral approximation decompositions for mixed-integer optimal control, 2018, Last accessed June 28, 2018 http://www.optimization-online.org/DB_HTML/2018/02/6472.html.