Journal of Global Optimization (2021) 80:523–550
https://doi.org/10.1007/s10898-021-01015-0

Partially distributed outer approximation

Alexander Murray[1] · Timm Faulwasser[1] · Veit Hagenmeyer[1] · Mario E. Villanueva[2] · Boris Houska[2]

Received: 9 May 2019 / Accepted: 20 March 2021 / Published online: 17 April 2021

Abstract
This paper presents a novel partially distributed outer approximation algorithm, named PaDOA, for solving a class of structured mixed integer convex programming problems to global optimality. The proposed scheme uses an iterative outer approximation method for coupled mixed integer optimization problems with separable convex objective functions, affine coupling constraints, and compact domain. PaDOA proceeds by alternating between solving large-scale structured mixed-integer linear programming problems and partially decoupled mixed-integer nonlinear programming subproblems that comprise much fewer integer variables. We establish conditions under which PaDOA converges to global minimizers after a finite number of iterations and verify these properties with an application to thermostatically controlled loads and to mixed-integer regression.

Keywords Mixed integer programming · Distributed optimization · Outer approximation · Global optimization

1 Introduction

A Mixed Integer Convex Program (MICP) is an optimization problem with convex objective and constraint functions, where the only non-convex constraint is that a subset of the opti-

✉ Alexander Murray
alexm3141@gmail.com

Timm Faulwasser
timm.faulwasser@tu-dortmund.de

Veit Hagenmeyer
veit.hagenmeyer@kit.edu

Mario E. Villanueva
meduardov@shanghaitech.edu.cn

Boris Houska
borish@shanghaitech.edu.cn

[1] Institute for Automation and Applied Computer Science, Karlsruhe Institute of Technology, Karlsruhe, Germany

[2] School of Information Science and Technology, ShanghaiTech University, Pudong, China

mization variables need to be integer-valued [8,36]. MICPs arise in a plethora of application areas ranging from AC transmission expansion planning and robust power flow problems [1,30], via thermal unit design and control [12], a variety of scheduling and layout design problems [53], design of multi-product batch plants [50], to obstacle avoidance and robotic motion planning problems [34].

Although MICPs are NP-hard in general, there exist a variety of algorithms for solving MICPs to global optimality [8]. State-of-the-art MICP solvers are based on tailored methods that exploit the fact that the integrality constraints are discrete while all other constraints are convex. Early attempts to develop tailored Branch and Bound (B&B) methods for MICP have been proposed in [24], mostly focussing on computational experiments and heuristics for selecting the branching variables and nodes. Improved versions of these early B&B methods for MICP can be found in [9,35]. Other early methods for solving MICP include generalized Benders decomposition methods [5,23], which are, however, less frequently used in state-of-the-art MICP solvers.[1]

Modern MICP implementations are often based on or related to Outer Approximation (OA), which goes back to Duran and Grossmann [16]. In contrast to B&B, OA alternates between solving Nonlinear Programs (NLP) with fixed integer values as well as Mixed Integer Linear Programs (MILP), which are constructed by linearizing the objective and constraint functions at the solutions of the NLP and which are used to update the integer variables. A notable extension of OA has been developed by Fletcher and Leyffer [20], who suggest to include curvature information in the relaxed integer program leading to a quadratic outer approximation method. A more recent approach to second order outer approximation is that of Kronqvist et al. [32]. Moreover, Kesavan and co-workers [28] have studied variants of OA for solving non-convex mixed integer problems. Another class of MICP methods are based on (extended) cutting plane methods [59] or combination of OA and branch-and-cut [49]; see also [56] for a general overview of polyhedral branch-and-cut methods. In recent years, there has been considerable progress in lift-and-project methods for MICP. An excellent overview and discussion of the state-of-the-art of such lift-and-project methods can be found in a recent article by Kilinç, Linderoth, and Luedtke [29].

Another recent trend in MICP solver development is the exploitation of separable structures by so-called extended formulations [25]. Here, the main idea is to introduce auxiliary variables in order to bound decoupled summands in additive expressions separately [58], which can lead to tighter polyhedral outer approximations. Such extended formulations have not only found their way into OA methods, as discussed in [25] and [33], but they can also be used to increase the performance of lift-and-project methods for MICP [29]. Also of note are several algorithms that exploit separability to decompose large-scale MINLP problems into MINLP subproblems based on block separability [41,42,45,46]. However, extended formulations only exploit separability to construct tighter outer approximations, but neither existing OA methods nor state-of-the-art lift-and-project methods ever attempt to break a large-scale MICP into decoupled MICPs with fewer integer variables. This is in contrast to distributed continuous convex optimization methods, such as dual decomposition [19,43], Alternating Direction Method of Multipliers (ADMM) [10,17,21], or Augmented Lagrangian based Alternating Direction Inexact Newton (ALADIN) methods [26], which can all be used to solve large-scale convex optimization problems to global optimality by alternating between solving small-scale convex optimization problems and sparse linear algebra operations. These methods typically require communication of the solutions of the decoupled problems between neighbours or to a central coordinator [10]. Although some researchers,

[1] For more details see [8].

[39,55], have attempted to apply these distributed local optimization methods in a heuristic manner, these methods cannot find global minimizers of non-convex problems reliably. This is due to the fact that ADMM, ALADIN, or similar distributed convex optimization method typically rely on strong duality results for augmented Lagrangians [52,54], which fail to hold in the presence of integrality constraints.

After reviewing extended formulations and related existing outer approximation methods in Sect. 2, the main contribution of this paper is presented in Sect. 3, which introduces a Partially Distributed Outer Approximation (PaDOA) method for MICPs with separable objective functions. In contrast to existing algorithms for structured MICP, PaDOA alternates between solving MICPs with fewer integer variables and large-scale MILPs. Section 3.5 discusses the global convergence properties of PaDOA, as summarized in Theorem 3. In this context, we additionally establish the fact that global optimality of a given feasible point of an MICP with N separable objectives and Nn optimization variables can be computationally verified by solving N partially-decoupled MICPs, each comprising at most n local integer variables, and one MILP with Nn integer variables. This result is summarized in Theorem 2, which analyzes one-step convergence conditions for PaDOA. In the sense that both MICPs as well as MILPs are NP hard in general [22,40], this result is not in conflict with existing complexity results for mixed integer optimization problems. However, there are solvers such as `CPLEX` [27], `Gurobi` [47], and many others [13], which are specifically designed for MIQPs and MILPs. Thus, the fact that one can reduce the task of verifying global optimality of a feasible point of a separable MICP with coupled affine constraints to the task of solving one MILP of a comparable size and several smaller subproblems, is—at least from a computational perspective—an important contribution. Notice that Sects. 2 and 3 focus on the theoretical properties of PaDOA, and consequently, a high level abstract notation is used to present these theoretical, yet, at this stage, conceptual ideas. These derivations lead to a formal prove of convergence, but, in general, the performance of PaDOA is problem dependent and a partial distribution of operations can only possibly lead to practical improvements if the problems are sufficiently structured. This aspect is illustrated by applying the method two MICP benchmark case studies in Sect. 4. The developments in this paper are not of pure theoretical interest only, but actually lead to a practical framework that has potential for future investigation, as outlined in our conclusions in Sect. 5.

1.1 Problem formulation

The present paper is concerned with mixed integer optimization problems of the form

$$\begin{aligned} V^\star = \min_{x\in X, z\in Z} \; & f(x,z) \\ \text{s.t.} \quad & Ax = b \end{aligned} \tag{1}$$

with separable objective function $f(x,z) = \sum_{i=1}^{N} f_i(x_i, z_i)$ and separable constraint sets

$$\begin{aligned} X = X_1 \times \ldots \times X_N \text{ with } \quad & X_i \subseteq \mathbb{R}^{n_i} \\ \text{and } \quad Z = Z_1 \times \ldots \times Z_N \text{ with } \quad & Z_i \subseteq \mathbb{Z}^{m_i} \; . \end{aligned} \tag{2}$$

The coupling matrix A and the vector b are assumed to be given. In this context, the following blanket assumption is used.

Assumption 1 The sets $X_1, X_2, \ldots, X_N$ are non-empty convex polytopes, the sets $Z_1, Z_2, \ldots, Z_N$ are non-empty and compact, and the functions f_i are convex on the convex hull of $X_i \times Z_i$.

The goal of this paper is to develop an efficient algorithm that finds ε-suboptimal points of (1), which are defined as follows:

Definition 1 A feasible point $(x^\star, z^\star) \in X \times Z$ with $Ax^\star = b$ is said to be an ε-suboptimal point of (1), with $\varepsilon > 0$, if

$$f(x^\star, z^\star) \leq f(x, z) + \varepsilon\,.$$

for all $(x, z) \in X \times Z$ with $Ax = b$.

Remark 1 The theoretical developments in this paper assume—for simplicity of presentation—that all coupled constraints are linear equations in x. This is not restrictive in the sense that coupled convex inequalities can always be reformulated by introducing slack variables [4,11,38].

Remark 2 Instead of (1), one could also consider more general optimization problems of the form

$$\begin{array}{ll} \min\limits_{x\in X, z\in Z} & f(x, z) \\ \text{s.t.} & Ax + Bz = b\,. \end{array} \tag{3}$$

However, under mild regularity assumptions [44], this problem is equivalent to

$$\begin{array}{ll} \min\limits_{x\in X, y\in \text{conv}(Z), z\in Z} & \sum\limits_{i=1}^{N} \{f_i(x_i, z_i) + \bar{\lambda}_i \|y_i - z_i\|_1\} \\ \text{s.t.} & Ax + By = b\,. \end{array} \tag{4}$$

with real-valued auxiliary variables y and L_1-penalty parameters $\bar{\lambda}_i \gg 0$. Here, conv (Z) denotes the convex hull of Z. Thus, for all theoretical purposes, it is sufficient to analyze problems of the form (1), where only the real-valued variables are coupled.

Remark 3 Modern MICP formulations, algorithms, and software can deal with rather general convex conic constraints [36]. Such constraints are left out for simplicity of presentation. Nevertheless all results in this paper can be easily extended for general conic constraints, as long as they are separable. From a purely theoretical perspective, one might argue that this can always be achieved by adding suitable convex penalty functions to the functions f_i, because this paper makes no assumptions on the differentiability properties of f. However, more tailored, practical algorithms that could exploit the structures of particular conic constraints are beyond the scope of this paper.

1.2 Notation

We use the notation

$$\partial_x g(x) = \left\{a \in \mathbb{R}^n \mid \forall y \in \mathbb{R}^n,\ g(y) \geq g(x) + a^\mathsf{T}(y - x)\right\}$$

to denote the set of subgradients of a convex function $g : \mathbb{R}^n \to \mathbb{R}$ with respect to the variable x.

2 Outer approximation

The following section reviews OA by using a slightly more abstract viewpoint that is based on modern convex analysis notation. As mentioned in the introduction, OA has been introduced in the 1980s by Duran and Grossmann [16] using different notation, but the following section presents the method in a more general setting—also including Hijazi's variants and extentions of OA—in order to prepare the developments in Sect. 3.

2.1 Polyhedral relaxations

In order to construct polyhedral outer approximations of the epigraph of the objective function f of (1), we consider the auxiliary optimization problem

$$f^\star(z) = \min_{x,y} \; f(x,z) \quad \text{s.t.} \quad \begin{cases} x = y & \mid \lambda \\ Ay = b \\ y \in X\,. \end{cases} \tag{5}$$

for a fixed integer parameter $z \in Z$. Here, x and y are real valued primal optimization variables and the notation "$x = y \mid \lambda$" is used to say that λ denotes the dual variable associated with the constraint $x = y$.

Proposition 1 *If Assumption 1 is satisfied, strong duality holds for (5), i.e., we have*

$$f^\star(z) = \max_{\lambda} \; \min_{x,y} \; f(x,z) + \lambda^T(y-x) \quad \text{s.t.} \quad \begin{cases} Ay = b \\ y \in X \end{cases} \tag{6}$$

for all $z \in Z$.

Proof See "Appendix 1". □

Let $x^\star(z)$, $y^\star(z)$, $\lambda^\star(z)$ denote any primal-dual solution of (5) in dependence on z. By writing out the optimality condition of (5) with respect to x, we find that

$$\lambda^\star(z) \in \partial_x f(x^\star(z), z),$$

i.e., $\lambda^\star(z)$ must be a subgradient of f at the optimal solution of (5). For the developments given below it is helpful to keep in mind that the reverse statement is not correct, i.e., a subgradient of f at $(x^\star(z), z)$ is not necessarily a dual solution of (5).

In contrast to the particular choice of the subgradient $\lambda^\star$ of f with respect to x, the construction of a subgradient of f with respect to z is less critical for the construction of outer approximation methods.[2] In the following, we assume that a function

$$\mu^\star(z) \in \partial_z f(x^\star(z), z),$$

is given, which returns a subgradient of f with respect to z at the optimal solution of (5). Because $f(x,z) = \sum_{i=1}^N f_i(x_i, z_i)$ is separable, the ith block components, $\lambda_i^\star(z)$ and $\mu_i^\star(z)$, of the subgradients of f are subgradients of f_i. Thus, the inequality

$$f_i(x_i, z_i) \geq f_i^\star(\hat{z}) + \left[\lambda_i^\star(\hat{z})\right]^\mathsf{T}(x_i - x_i^\star(\hat{z})) + \left[\mu_i^\star(\hat{z})\right]^\mathsf{T}(z_i - \hat{z}_i)$$

[2] Note that we are referring to the relative importance of z and λ and that the choice of subgradient should in general not be neglected. For example, [18] shows that with a poor choice of subgradients OA can fail to converge for convex nonsmooth MINLP problems.

holds for all $x_i \in X_i$, and $z_i \in Z_i$ and all $\hat{z} \in Z$. Here, the shorthand

$$f_i^\star(\hat{z}) = f_i(x_i^\star(\hat{z}), \hat{z}_i)$$

is used. In this context, it is important to notice that the function $f_i^\star(\hat{z})$ depends on the whole vector $\hat{z}$, not only on its ith component, $\hat{z}_i$, because the equality constraints in (6) introduce a non-trivial coupling.

More generally, if $\Pi \subseteq Z_i$ denotes finite set of points in Z, we associate with Π a set of hyperplane coefficients

$$\mathcal{H}_i(\Pi) = \left\{ (\alpha, \beta, \gamma) \;\middle|\; \begin{array}{l} z \in \Pi \\ \alpha = \lambda_i^\star(z) \\ \beta = \mu_i^\star(z) \\ \gamma = f_i^\star(z) - \alpha^\mathsf{T} x_i^\star(z) - \beta^\mathsf{T} z_i \end{array} \right\} . \tag{7}$$

Notice that this set of hyperplane coefficients defines a polyhedral outer approximation of the epigraph of f_i. Thus, these coefficients can be used to construct a piecewise affine lower bound on f_i, which is for all $(x_i, z_i) \in X_i \times Z_i$ given by

$$\Phi_i(x_i, z_i, \Pi) = \max_{(\alpha,\beta,\gamma)\in H_i(\Pi)} \left\{ \alpha^\mathsf{T} x_i + \beta^\mathsf{T} z_i + \gamma \right\} . \tag{8}$$

Finally, we can construct the function

$$\Phi(x, z, \Pi) = \sum_{i=1}^{N} \Phi_i(x_i, z_i, \Pi) .$$

This function is—by construction—a piecewise affine lower bound on f,

$$\forall (x, z) \in X \times Z, \qquad \Phi(x, z, \Pi) \leq f(x, z) . \tag{9}$$

Next, our particular choice of the subgradient $\lambda^\star(z)$ of f as the dual solution of (5) enables us to establish the following tightness property of the affine lower bound Φ.

Lemma 1 *Let $\Pi \subseteq Z$ be any finite set of points. If Assumption 1 holds, then*

$$f^\star(z) = \min_{x\in X} \; \Phi(x, z, \Pi) \quad \text{s.t.} \quad Ax = b . \tag{10}$$

holds for all $z \in \Pi$.

Proof See "Appendix 2". □

Note that Lemma 1 can be viewed as a special case of Theorem 2.3 in [23].

Remark 4 If the set Π consists of m points, the computational cost for constructing the lower bound (9) of f has order $\mathbf{O}(mN)$. Notice that if we would have ignored the separable structure of f, the computational cost of computing the same lower bounding function would have been of order $\mathbf{O}(m^N)$. Thus, the construction of (9) as a sum of the lower bounds of the separable objective function is much cheaper than a direct construction of lower bounds of f. This reduction in complexity has for the first time been observed and exploited by Hijazi and co-workers [25]. By now, the expoitation of separability via extended formulations can be considered as a standard that has been adopted in many modern MICP algorithms and software tools [29,36,37].

2.2 Outer approximation algorithm

Algorithm 1 outlines the main steps of the outer approximation algorithm. Notice that this algorithm basically coincides with the original outer approximation algorithm that has been proposed in [16]. The only notable differences of Algorithm 1 compared to traditional OA are that the MILP in Step 3 uses the extended formulation based outer approximation variant from [25]. Moreover, because we do not assume that f is differentiable, we have to use the particular choice, $\lambda^\star(z)$, of the subgradient, which is found as the dual solution[3] of (5).

Algorithm 1: Outer Approximation for MICP

Input: Initial guess $z \in Z$ and a numerical tolerance $\varepsilon > 0$.
Initialization: Set $\Pi = \varnothing$ and $U = \infty$.
Repeat:

1. Solve the convex optimization problem

$$f^\star(z) = \min_{x,y} \; f(x,z) \quad \text{s.t.} \quad \begin{cases} x = y & \mid \lambda \\ Ay = b \\ y \in X\,. \end{cases} \tag{11}$$

2. If (11) has no feasible solution, return a certificate of infeasibility and terminate. Otherwise, update

$$U \leftarrow \min\left\{U,\; f^\star(z)\right\} \quad \text{and} \quad \Pi \leftarrow \Pi \cup \{z\}\,.$$

3. Solve the (extended) MILP

$$(x^+, y^+, z^+) \in \underset{x \in X, y, z \in Z}{\operatorname{argmin}} \sum_{i=1}^{N} y_i \quad \text{s.t.} \quad \begin{cases} \forall i \in \{1,\ldots,N\}, \\ \forall (\alpha_i, \beta_i, \gamma_i) \in \mathcal{H}_i\,(\Pi) \\ \alpha_i^\mathsf{T} x_i + \beta_i^\mathsf{T} z_i + \gamma_i \le y_i \\ Ax = b \end{cases} \tag{12}$$

4. If $U - \sum_{i=1}^N y_i^+ \le \varepsilon$, terminate.
5. Update $z \leftarrow z^+$ and go to Step 1.

Notice that Step 1 of Algorithm 1 solves (1) under the additional constraint that the integer z is fixed. This implies that

$$f(x^\star, z) \ge V^\star$$

is an upper bound on the optimal objective value $V^\star$ of (1). Thus, the current upper bound U can be updated in Step 2. Moreover, the MILP (20) is by construction a relaxation of (1) which implies that the solution of Step 3, $\sum_{i=1}^N y_i^+$, is a lower bound on the optimal objective value $V^\star$. Thus, the difference, $U - \sum_{i=1}^N y_i^+$, between the current upper and lower bounds can be used as a termination criterion, which is implemented in Step 3 of Algorithm 1. The

[3] The idea to use dual solutions as subgradients for the construction of polyhedral outer approximation is not new and can—in a very similar setting—be found in [36].

following finite termination result for outer approximation is (at least in very similar versions) well-known in the literature [16,36].

Theorem 1 *If Assumption 1 is satisfied, then Algorithm 1 terminates after a finite number of iterations.*

Proof See "Appendix 3". □

Remark 5 Algorithm 1 uses Hijazi's extended formulation [25] for constructing the MILPs (20), which arguably exploits separability of the objective function to some extent. However, Algorithm 1 is not a fully distributed algorithm. In fact, a major disadvantage of Algorithm 1 becomes apparent, if one considers the special case that $A = 0$ and $b = 0$. In this case, the optimal solution of (1) could have been found with much less effort by solving the separable MICPs,

$$\min_{x_i \in X_i, z_i \in Z_i} f_i(x_i, z_i)$$

which have much fewer integer variables. However, if Algorithm 1 is applied to such a problem with redundant equality constraint, this property is not detected and a large number of large-scale NLPs and large scale MILPs might have to be solved instead, until convergence is achieved. The goal of this paper is to mitigate this limitation of Algorithm 1 by proposing a partially distributed outer approximation algorithm that exploits the structure of the separable objective in a better way.

3 Partially distributed outer approximation algorithm

This section introduces a partially distributed outer approximation optimization algorithm for finding ε-suboptimal solutions of (1).

3.1 Partially decoupled upper bounds

The main idea of many distributed convex and local optimization methods is to solve a set of smaller-scale decoupled optimization problems in place of a single large one [5,10,14]. Similarly, consider partially decoupled optimization problems of the form

$$V_k(z) = \min_{x,y,\zeta_k} \; f_k(x_k, \zeta_k) + \Psi_k(x, z)$$
$$\text{s.t.} \quad \begin{cases} x = y & \mid \lambda \\ Ay = b \\ y \in X \\ \zeta_k \in Z_k \end{cases} \tag{13}$$

for $k \in \{1, \ldots, N\}$. Note that Problem (13) regards only the local integer variables $\zeta_k \in Z_k$ as optimization variables, while all other integers are fixed. However, concerning the real-valued variables, the whole vector $x \in X$ is kept as an optimization variable. In this context, the shorthands

$$\forall (x, z) \in X \times Z, \qquad \Psi_k(x, z) = \sum_{j \neq k} f_j(x_j, z_j)$$

are introduced in order to keep the x-dependence of the remaining summands, i.e., all objective terms whose index is not equal to k. As in the previous section, λ denotes the dual solution that is associated with the consensus constraint "$x = y$". Because the constraint $\zeta_k \in Z_k$ enforces integrality, strong duality of (13) does not hold in general. However, if $\zeta_k^\star(z)$ denotes an optimal solution of (13) for the integer variable and if Assumption 1 holds, we still have

$$V_k(z) = \max_{\lambda} \min_{x,y} \quad f_k(x_k, \zeta_k^\star(z)) + \Psi_k(x, z) + \lambda^\mathsf{T}(y - x)$$
$$\text{s.t.} \quad \begin{cases} Ay = b \\ y \in X \,. \end{cases} \tag{14}$$

The proof of this statement is completely analogous to Proposition 1, i.e., if the linear coupling constraint $Ax = b$ has a solution in X, a maximizer of (14) exists and can be used to define a suitable subgradient. Also note that the functions V_k yield upper bounds on the objective value of (1),

$$\forall z \in Z, \qquad \min_k \; V_k(z) \geq V^\star \tag{15}$$

At this point, it should be mentioned that one basic assumption of the algorithmic developments in this paper is that the complexity of the mixed integer optimization problems of interest depends mostly on the number of integer variables. This is in contrast to the number of real-valued variables, which may be assumed to have a negligible influence on the overall complexity of the mixed-integer optimization problem. In other words, we assume that (13) is much easier to solve than (1) in the sense that it contains much fewer integer variables, although both problems have the same number of real-valued variables. Here, it is important to keep in mind that, although the algorithmic developments in this paper are inspired by the field of distributed optimization, the algorithm in this paper is (at least in the form in which we present and analyze it) not fully distributed. This is because solving (13) requires the evaluation of the function Ψ_k, which, in turn, requires the evaluation of all functions f_j with $j \neq k$.

3.2 Partially decoupled lower bounds

In this paper, we suggest to solve the decoupled MICPs (13) by lower level solvers that implement the traditional outer approximation algorithm that has been reviewed in Sect. 2.2. Notice that if Assumption 1 is satisfied, strong duality holds, i.e., these lower level solvers will return piecewise affine models

$$\Theta_k^\star : X \times Z_k \to \mathbb{R},$$

which must satisfy the condition

$$V_k(z) - \epsilon_\mathrm{L} \;\leq \min_{x \in X, \zeta \in Z_k} \Theta_k^\star(x, \zeta) \quad \text{s.t.} \quad Ax = b \tag{16}$$

upon termination. Here, $\epsilon_\mathrm{L} \geq 0$ denotes the numerical tolerance of the lower level OA solvers. Notice that the optimization problem on the right hand of (16) corresponds to the last MILP relaxation that is solved by the lower level OA solver. In practice the function $\Theta_k^\star$ can be stored by maintaining a set of hyperplane coefficients as explained in detail in the previous section.

The main idea of partially distributed outer approximation is to communicate the piecewise lower bounding functions $\Theta_k^\star$ to a central coordinator, who constructs a piecewise affine lower

bound on the function f, solves a master MILP problem, and updates z. Here, one option is use the maximum over the function $\Theta_k^\star$ in order to obtain the lower bound

$$\forall x \in X,\ \forall z \in Z, \qquad \max_k\ \Theta_k^\star(x, z_k)\ \leq\ f(x, z)\ . \tag{17}$$

However, in order to arrive at a practical implementation, it is recommendable to further refine this bound. This can be done by maintaining a collection of integers, $\Pi \subseteq Z$, such that the function

$$\Theta(x, z) = \max\left\{\ \Phi(x, z, \Pi),\ \max_k\ \Theta_k^\star(x, z_k)\ \right\}, \tag{18}$$

can be used as a piecewise affine lower bound on f. Recall that the function Φ, which has been introduced in the previous section, exploits the separability properties of f. The integer collection Π is then maintained by updating

$$\Pi \leftarrow \Pi \cup \{\zeta^\star\},$$

where $\zeta^\star = [\zeta_1^\star, \zeta_2^\star, \ldots, \zeta_N^\star]$ is an integer vector, whose components are optimal solutions for the integer variables of the partially decoupled problems (13).

3.3 Partially distributed outer approximation (PaDOA)

Algorithm 2 outlines a partially distributed algorithm for solving (1). There are four main steps. In the first step, the partially decoupled MICPs of the form (13) are solved by using a traditional outer approximation method. Under the assumption that the original MICP (1) is feasible, the partially decoupled MICPs are feasible, too. Thus, the outer approximation solvers will return optimal integer solutions $\zeta_k^\star$ and associated piecewise affine lower bounds $\Theta_k^\star$ such that (16) is satisfied. The second step of Algorithm 2 updates the associated upper bound U based on the inequality (15) as well as the piecewise affine lower bound. In practice, this step is implemented by storing the union of all supporting hyperplane coefficients that are needed to represent Θ. Finally, the third step of Algorithm 2 solves a large scale MILP problem. This MILP is constructed in analogy to the corresponding step in the traditional outer approximation algorithm. It yields a lower bound,

$$\Theta(x^+, z^+) \leq V^\star,$$

on the objective value $V^\star$ of (1). Thus, the difference between the current upper and lower bounds,

$$U - \Theta(x^+, z^+),$$

can be used as a termination criterion, which is implemented in the fourth step of Algorithm 2. If the termination is not successful, the integer variables z are updated, and the algorithm subsequently proceeds to the next iteration.

Notice that the main difference between Algorithms 1 and 2 is the introduction of partially decoupled MICP problems that can be solved separately and which contain much fewer integer variables than the original MICP (1). The theoretical results in Sect. 3.5 will elaborate further on the benefits of this alternation strategy. Moreover, in Sect. 4 a numerical case study is examined, which illustrates the practical advantages of Algorithm 2.

Remark 6 For the special case that the functions f_i are convex and piecewise affine, the original optimization problem is an MILP, which could also be solved directly by using an

Algorithm 2: Partially Distributed Outer Approximation (PaDOA)

Input: Initial guess $z \in Z$ and a numerical tolerance $\varepsilon > 0$.
Initialization: Set $\Pi = \varnothing$, $\Theta(\cdot,\cdot) = -\infty$, and $U = \infty$.
Repeat:

1. Solve for all $k \in \{1, \dots, N\}$ the partially decoupled MICPs

$$V_k(z) = \min_{x,y,\zeta_k} \; f_k(x_k, \zeta_k) + \Psi_k(x, z) \qquad \text{with} \qquad \Psi_k(x, z) = \sum_{j \neq k} f_j(x_j, z_j) \; .$$
$$\text{s.t.} \quad \left\{ \begin{array}{ll} x = y & \mid \lambda \\ Ay = b & \\ y \in X & \\ \zeta_k \in Z_k & \end{array} \right. \tag{19}$$

   If (19) is infeasible, terminate and return a certificate of infeasibility. Otherwise, update the set $\Pi \leftarrow \Pi \cup \{\zeta_k^\star\}$ and construct a piecewise affine model Θ_k^* such that condition (16) is satisfied.

2. Update the upper bound $U \leftarrow \min\{U, \; V_1(z), \dots, V_N(z)\}$ and construct the piecewise lower bounding function $\Phi(x, z, \Pi)$ as in (8).

3. Update the lower bound

$$\forall x \in X, \; \forall z \in Z, \qquad \Theta(x, z) \leftarrow \max \left\{ \; \Theta(x, z) \; , \; \Phi(x, z, \Pi) \; , \; \max_k \; \Theta_k^\star(x, z_k) \; \right\}$$

4. Solve the MILP problem

$$(x^+, z^+) \in \operatorname*{argmin}_{x \in X, z \in Z} \Theta(x, z) \quad \text{s.t.} \quad Ax = b \tag{20}$$

5. If $U - \Theta(x^+, z^+) \leq \varepsilon$, terminate. Otherwise, update $z \leftarrow z^+$ and go to Step 1.

MILP solver. If one uses Algorithm 2 instead, this algorithms solves an MILP by solving a sequence of MILPs—a strategy that may seem rather counter-intuitive on the first view. However, one interesting observation of this paper is that Algorithm 2 can even for (sufficiently structured) MILPs help to speed up the convergence process, because the complexity of the MILPs in Step 4 of Algorithm 2 depends on the complexity of the current lower bound Θ and eventually is much easier to solve than the original MILP. This effect is illustrated by numerical experiments in Sect. 4.2.

3.4 Relation to distributed local optimization methods

The idea to "augment" the local objective functions f_i with a suitable function Ψ_i is frequently used in the context of distributed local optimization algorithms. For example, in the context of dual decomposition [19,43], one augments the separable functions f_i with linear functions of the form[4]

$$\Psi_i(x) = \sigma^\mathsf{T} A x,$$

[4] In the context of convex optimization, the functions Ψ_i only depend on x, and Problem (1) is considered without integer variables z.

where σ is the current dual iterate. Similarly, in the context of ADMM or ALADIN, one uses augmented Lagrangians [3,48], as in

$$\Psi_i(x, y) = \sigma^\mathsf{T} A x + \frac{\rho}{2}\|x - y\|^2 , \tag{21}$$

where z and σ are the current primal and dual iterates; see [10,17,26]. In fact, the construction of Algorithm 2 is inspired by the distributed local nonlinear programming method ALADIN. Here, we recall that ALADIN alternates between solving small-scale decoupled NLPs that are augmented by terms of the form (21) and large scale equality constrained quadratic programming problems that update σ and y [26]. This is in analogy to Algorithm 2, which alternates between solving decoupled MICPs (Step 1) and large-scale coupled MILPs (Step 3). However, unlike ALADIN, augmented Lagrangians are not used in Algorithm 2 as Lagrange multipliers in integer programming are not related to sensitivity and generally not applicable.

Also note that the construction of the functions Ψ_i in Algorithm 2 also has similarities with Gauss-Seidel or more general block-coordinate descent methods [57,60] in the sense that a partial decoupling is obtained by fixing some of the integer variables while others are optimized. However, despite all these analogies and similarities of Algorithm 2 with methods from the field of local and convex optimization, we would like to highlight that all these existing distributed optimization methods are not reliably applicable to Problem (1) [55].

3.5 Convergence analysis

In this section we provide a concise overview of the convergence properties of Algorithm 2. The following theorem establishes one of the main results of this paper, namely, that Algorithm 2 converges after one iteration if the integer iterate, z, is initialized with an optimal solution of (1). This is in contrast to Algorithm 1, which does not necessarily terminate after a small number of steps—not even if it is initialized at an optimal solution.

Theorem 2 *Let Assumption 1 be satisfied and let $(x^\star, z^\star)$ be a minimizer of (1). If Algorithm 2 is initialized with $z = z^\star$ and if the termination tolerances of the lower level solvers satisfy $\epsilon_\mathrm{L} \leq \epsilon$, then the termination criterion in Step 4 is satisfied. In other words, the algorithm terminates after one step.*

Proof See "Appendix 4". □

Notice that the statement of the above theorem is of fundamental relevance and a very favorable property of PaDOA. If we work with other global optimization methods, say branch-and-bound, an empirical observation is that such existing global optimization algorithm often find a global solution early on but then keep on iterating until the lower bound is accurate enough to prove global optimality. In contrast to this, PaDOA terminates as soon as a global minimizer is added to the collection Π. In fact, Theorem 2 implies that global optimality of a point $z^\star \in Z$ can be verified by solving the N instances of the partially decoupled MICPs and the master MILP (20). Notice that this result is not in conflict with existing results from the field of complexity theory, because the master MILP (20) remains NP-hard [22,40].

Remark 7 The result of Theorem 2 relies heavily on the convexity of the functions f_i on the convex hull of $X_i \times Z_i$, although this fact is not highlighted explicitly in the proof. This convexity assumption is first of all required implicitly by our assumption that the lower level

solvers return piecewise level models Θ_k, which satisfy the termination condition (16) (this assumption is only reasonable if strong duality holds) and which need to be global lower bounds on f. These properties are in general all not satisfied if one considers more general non-convex MINLPs.

The following theorem establishes the fact that Algorithm 2 converges after a finite number of iterations under exactly the same conditions under which convergence of Algorithm 1 can be established.

Theorem 3 *Let Assumption 1 be satisfied. If the termination tolerances of the lower level solvers satisfy $\epsilon_{\mathrm{L}} \leq \epsilon$, then Algorithm 2 terminates after a finite number of steps (independently of the initialization).*

Proof See "Appendix 5". □

4 Implementation and case study

This section presents two benchmark problems to assess the computational performance of Algorithm 2. The first aims to provide a cost-optimal heater/cooler activation scheme such that a set of temperatures remain within pre-set limits. The objective is formulatable as an MILP, MIQP, or higher order MICP allowing for testing on a broad class of problems. Furthermore, each problem instance is scalable both vertically (in time horizon/subproblem size) and horizontally (number of regions/number of subproblems), which allows for an examination of the benefit which may come from the parallelizable Step 1 of Algorithm 2.

A second benchmark example comes in the form of a mixed-integer lasso problem. This is used to demonstrate the versatility of Algorithm 2 and verify the results of the first benchmark problem given the presence of an L_1-norm. Like the first benchmark problem, it is also scalable both vertically and horizontally, and the results for ten unique instances of the problem are given. The Partially Distributed Outer Approximation method is implemented in MATLAB R2017b. The optimization subproblems are solved using Gurobi [47] implemented via CasADi v1.9.0 [2]. All numerical experiments were run on a 2.9 GHz Intel Core i5-4460S CPU with 8 GB of RAM.

4.1 Thermostatically controlled loads

An important problem in the planning and operation of a heating and/or cooling system is the scheduling of so-called Thermostatically Controlled Loads (TCLs) [31]. These are devices that are used to regulate the temperature of a room/building within a certain user-defined interval known as a "deadband". The optimal operation strategy is especially difficult to determine when a non-constant cost function is introduced for a population of heterogeneous TCLs [61]. The cost function may represent the cost of electricity or user-discomfort from noise generation. Regardless, such devices typically only have an "on" and an "off" setting and thus the resulting scheduling problem can be formulated as a binary MIP for R regions with a discretized time horizon H, consisting of hour-long time steps.

$$\min_{T(\cdot),u(\cdot)} \sum_{i=1}^{R} \sum_{t=0}^{H-1} c(t)u(t) + \gamma(T_i(t) - T_{i,ref}(t))^2, \tag{22a}$$

$$\text{subject to} \quad \forall i \in \{1, \ldots, R\},$$

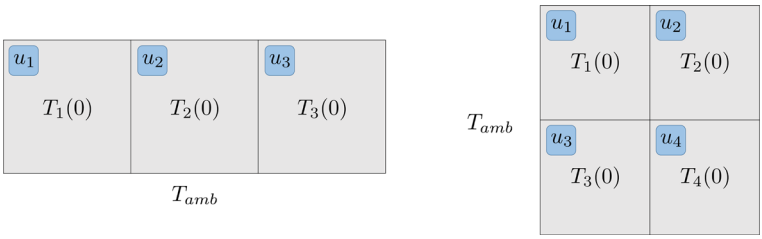

Fig. 1 Two room configurations with controlled cooling elements u_i, ambient temperature T_{amb} and initial temperatures $T_i(0)$

$$\underline{T}_i \leq T_i(t) \leq \overline{T}_i, \quad \forall t \in \{0, \ldots, H\} \tag{22b}$$

$$u_i(t) \in \{0, 1\}, \qquad \forall t \in \{0, \ldots, H-1\} \tag{22c}$$

$$\forall t \in \{0, \ldots, H\},$$

$$T_i(t+1) = T_i(t) + b_i u_i(t) + a_i \left(\frac{T_i(t) + T_{amb}(t) + \sum_{j \in N(i)} T_j(t)}{|N(i)| + 2} - T_i(t) \right), \tag{22d}$$

where $c(t)$ is the vector of device costs at time t, γ is a comfort parameter, $\underline{T}_i$ and $\overline{T}_i$ are the deadband temperature limits of device i, a_i and b_i are heat transfer parameters, $T_{amb}(t)$ is the ambient temperature at time t and $N(i)$ are the number of regions neighbouring i. Equation (22d) models the thermodynamics of each room in a simplified manner, i.e., it takes an average of the current and surrounding temperatures to update the temperature of the next time step. This formulation results in $H+1$ real-valued and H binary variables per region. Figure 1 shows two possible initial configurations of (22). The ambient temperature is taken from [15] for two days in June 2017 in the Karlsruhe (Germany) area, with a reference temperature $T_{i,ref} = 20^{\text{deg}}$ for each room. High prices of 25.67 cents/kWh are set from 2pm to 8pm (time steps 6 to 12 and 29 to 35) with low and medium prices of 2.46 cents/kWh and 4.62 cents/kWh in all other time steps. Each region is initialized at 20 degrees with $a_i = 0.2$ and $b_i = -2$. As the temperature for each room is more influenced by past temperatures than neighbouring rooms, Problem (22) is partitioned spatially for Algorithm 2, with each room in its own partition.

4.2 Results for MILP

If the comfort parameter γ is taken to be zero then Problem (22) is linear and separable but coupled in both its discrete and real-valued variables. Shown in Tables 1 and 2 are the simulation results for each configuration, respectively. The results of Algorithm 2 are compared with results obtained from the MIP solver Bonmin with the Branch and Bound (B-BB) and Outer Approximation (B-OA) algorithm settings [7] as well as the commercial MIQP solvers Gurobi and CPLEX [27]. An example solution for the 3 room case is depicted in Fig. 2.[5]

At first glance, the results from Tables 1 and 2 may seem surprising since the 4 room case has more space to keep cool but nonetheless is able to do so at a lower cost than the 3 room case. This is due to an insulation effect that the 4 room configuration enjoys. With the activation of two coolers in the first six time steps, the room temperatures can stay within their

[5] Only two temperature trajectories are visible due to both end-rooms having identical solutions.

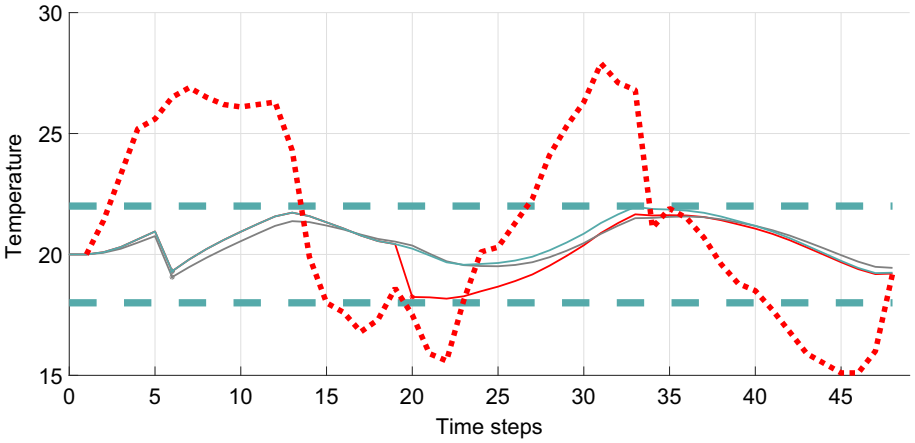

Fig. 2 The red dotted line is the ambient temperature, the blue dotted lines are the limits of the temperature deadzone and the solid lines are the temperature trajectories of each region in the three room scenario. Highlighted on the trajectories are points where the coolers are activated. (Color figure online)

Table 1 Results obtained for the TCL problem with a 3-room configuration

Time steps	8	24	48	62
Alg. 2				
Obj.	0	13.86	16.32	21.23
Time (s)	0.15	0.20	0.89	5.23
Iter.	2	1	2	2
B-OA				
Obj.	0	13.86	16.32	21.23
Time (s)	0.15	31.06	1,345	52,395
B-BB				
Obj.	0	13.86	16.32	21.23
Time (s)	0.15	29.19	480.27	828.08
Gurobi				
Obj.	0	13.86	16.32	21.23
Time (s)	0.22	0.56	0.85	4.39
CPLEX				
Obj.	0	13.86	16.32	21.23
Time (s)	0.07	0.16	0.54	2.81

deadbands for the entire 48 h period. In contrast, the 3 room configuration is more susceptible to the ambient temperature and requires more use of the coolers. This also seems to have increased the computational complexity of the problem and requires more time for the 3 room case to be solved than the 4 room case. It should be noted that several initializations were tested and the solution times were not significantly affected, implying that this was not the cause of the runtime differences in the two cases.

One of the advantages of using a distributed method is the ability to solve problems that would be otherwise intractable for a centralized solver. Tables 1 and 2 show results for cases containing up to 496 variables, but even larger problems may be considered. Table 3

Table 2 Results obtained for the TCL problem with a 4-room configuration

Time steps	8	24	48	62
Alg. 2				
Obj.	0	9.24	9.24	11.69
Time (s)	0.17	0.25	0.39	1.41
Iter.	2	2	2	2
B-OA				
Obj.	0	9.24	9.24	11.69
Time (s)	0.14	28.34	48.42	1,610
B-BB				
Obj.	0	9.24	9.24	11.69
Time (s)	0.18	33.11	56.83	822.48
Gurobi				
Obj.	0	9.24	9.24	11.69
Time (s)	0.22	0.37	0.51	1.08
CPLEX				
Obj.	0	9.24	9.24	11.69
Time (s)	0.07	0.09	0.11	0.48

Table 3 Results obtained for the TCL problem with a linear room configuration

Rooms	7	7	7	10	12	18	20
Alg. 2							
Time steps	8	24	48	48	36	24	24
Obj.	0	23.1	28.02	37.26	41.88	50.82	55.44
Time (s)	0.15	1.03	4533.4	9127.5	8294	853.7	16,658
Iter.	2	2	2	2	2	2	2
B-OA							
Obj.	0	23.1	N/A	N/A	N/A	N/A	N/A
Time (s)	0.16	5134	N/A	N/A	N/A	N/A	N/A
B-BB							
Obj.	0	23.1	28.02	37.26	41.88	50.82	55.44
Time (s)	0.18	690.23	1682.5	3076.0	3261.5	2999.1	4658.4
Gurobi							
Obj.	0	23.1	28.02	N/A	41.88	50.82	N/A
Time (s)	0.25	0.93	56.09	N/A	56,842	31,926	N/A
CPLEX							
Obj.	0	23.1	28.02	37.26	41.88	50.82	N/A
Time (s)	0.08	0.65	150.31	13,416	12,885	57,108	$> 245{,}000$

Table 4 Results obtained for Problem (22) with a 3-room configuration

Time steps	8	24	48	62
Alg. 2				
Obj.	17.83	60.15	104.07	134.43
Time (s)	0.24	0.49	1.36	3.17
Iter.	4	3	3	3
B-OA				
Obj.	17.83	60.15	104.07	134.43
Time (s)	4.36	46.94	399.49	1702.60
B-BB				
Obj.	17.83	60.15	104.07	134.43
Time (s)	9.09	70.40	502.76	1152.90
Gurobi				
Obj.	17.83	60.15	104.07	134.43
Time (s)	0.44	0.50	0.66	0.83
CPLEX				
Obj.	17.83	60.15	104.07	134.43
Time (s)	0.19	0.27	0.25	0.54

shows results for a variety of time horizons and rooms. Here, the room configuration is instead arranged such that the rooms are in a line. While unrealistic for most buildings, this setup is realistic for the temperature control of a train or rooms next to a corridor. Mathematically, this example differs somewhat from the other two. While the other problems have a significant amount of coupling between the control variables, this is not the case for the linear room configuration. The sparsity induced by this problem structure allows Algorithm 2 to outperform both Gurobi and CPLEX (applied to the centralized problem).

4.3 Results for MIQP

If the comfort parameter γ is larger than zero then Problem (22) becomes a convex MIQP.[6] As in Sect. 4.2, the results of Algorithm 2 for each room configuration are compared with those obtained from Bonmin, Gurobi, and CPLEX. The value of γ was chosen to be one to allow for an equal weighting of comfort and cost. These results are displayed in Tables 4, 5, and 6.

Shown in Fig. 3 are the trajectories obtained for the three-room scenario with a temperature deviation penalization. In contrast to Fig. 2, a quadratic penalty term is used to model discomfort caused by deviations from the set temperature. Indeed, the solution with $\gamma = 1$ yields a trajectory with a similar number of activations as when $\gamma = 0$ but with temperature trajectories that stay much closer to the middle of the deadband.

[6] Convex in the same notion of convexity in MICPs. That is, a problem where the continuous relaxation yields a convex quadratic program.

Table 5 Results obtained for Problem (22) with a 4-room configuration

Time steps	8	24	48	62
Alg. 2				
Obj.	21.68	52.26	99.68	134.43
Time (s)	0.37	0.65	8.82	31.07
Iter.	5	3	3	4
B-OA				
Obj.	21.68	52.26	99.68	134.43
Time (s)	14.61	6.78	613.95	7,024.10
B-BB				
Obj.	21.68	52.26	99.68	134.43
Time (s)	30.39	11.13	661.95	1,495.30
Gurobi				
Obj.	21.68	52.26	99.68	134.43
Time (s)	0.41	0.54	0.72	2.31
CPLEX				
Obj.	21.68	52.26	99.68	134.43
Time (s)	0.15	0.16	0.41	2.15

Table 6 Results obtained for Problem (22) with a linear 7 room configuration

Time steps	8	24	48	62
Alg. 2				
Obj.	39.50	112.08	201.19	267.04
Time (s)	1.57	2.55	260.2	1022.38
Iter.	4	3	4	3
B-OA				
Obj.	39.50	112.08	N/A	N/A
Time (s)	358.12	1833.9	N/A	N/A
B-BB				
Obj.	39.50	112.08	201.19	267.04
Time (s)	593.17	955.77	6095.6	15,582
Gurobi				
Obj.	39.50	112.08	201.19	267.04
Time (s)	0.95	0.72	10.83	65.59
CPLEX				
Obj.	39.50	112.08	201.19	267.04
Time (s)	0.93	0.77	6.53	17.50

4.4 Higher Order Convex Problems

One of the advantages of the proposed algorithm is that it is applicable to a relatively large class of problems (namely, MICPs). While Sect. 4.3 shows favourable results for both Gurobi and CPLEX, if the problem were adjusted slightly such that it were no longer an MIQP then these solvers would no longer be applicable. For example, if the objective function of Problem

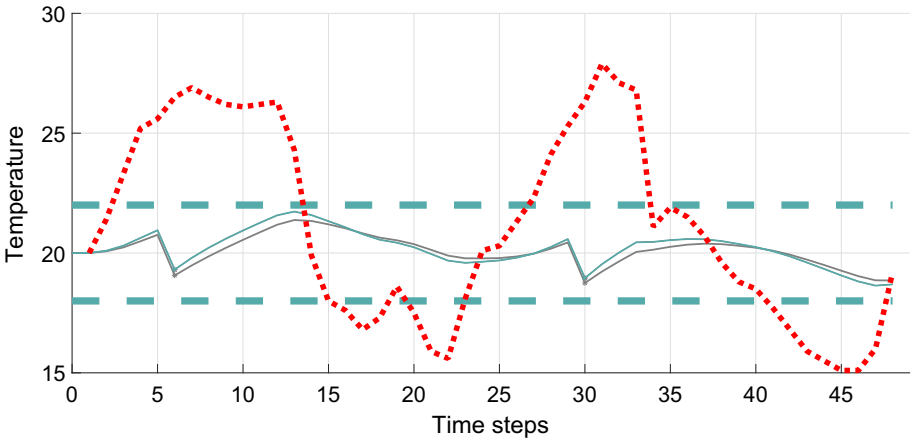

Fig. 3 The three room scenario, with the comfort parameter $\gamma = 1$. The red dotted line is the ambient temperature, the blue dotted lines are the limits of the temperature deadzone and the solid lines are the temperature trajectories of each region in the three room scenario. Highlighted on the trajectories are points where the coolers are activated. (Color figure online)

(22) became

$$\min_{T(\cdot),u(\cdot)} \sum_{t=0}^{H-1} c(t)u(t) + \gamma(T_i(t) - T_{ref}(t))^4,$$

then this would still be solvable via PaDOA, but not Gurobi or CPLEX. However, Bonmin can still be applied.[7] The results for a variety of such problem configurations are shown below in Table 7. Therein it can be observed that Algorithm 2 returns the same, global solution as Bonmin with the B&B sub-algorithm, and does so in less time. The runtime difference is particularly striking for the 7 room scenarios as these contain the most variables and have the greatest potential for parallelization.

4.5 Discussion of TCL Benchmark Results

Algorithm 2 is able to converge quite quickly for the MILP instance of (22) since it is underapproximating a linear problem with linear functions and thus can prove optimality quickly and with few iterations. More supporting hyperplanes are needed in the MIQP case and thus we see a couple more iterations and a slightly longer runtime for Algorithm 2. As shown in Figs. 4 and 5, the upper and lower bound converge quite quickly for the MIQP and MINLP cases, but require several iterations to refine the solution.

Interestingly, the MIQP cases is generally obtained more quickly than the MILP case, even though one might expect the higher order problem to be more difficult to solve. The quadratic term should make the MIQP instance of Problem (22) more computationally difficult than the MILP instance, but in some cases both Gurobi and CPLEX actually require less time. It may be the case that the MILP instance of (22) is more poorly conditioned than the MIQP instance, and thus the centralized solvers require much more effort to guarantee the optimality of a

[7] It should be noted that all results seen in this section for Algorithm 2 use Bonmin to solve the MICP subproblems.

Table 7 Results obtained for Problem (22), but with a 4th order objective function

Rooms	Time steps	Alg. 2			B-B&B		B-OA	
		Obj.	Time (s)	Iter.	Obj.	Time (s)	Obj.	Time (s)
3	8	16.72	4.20	5	16.72	7.51	N/A	N/A
3	24	76.86	19.14	4	76.86	308.64	N/A	N/A
3	48	115.62	95.56	3	115.62	547.98	N/A	N/A
3	62	141.44	405.19	6	141.44	779.41	N/A	N/A
4	8	21.73	4.11	6	21.73	16.93	N/A	N/A
4	24	44.94	6.47	3	44.94	9.93	N/A	N/A
4	48	86.72	151.58	5	86.72	477.60	N/A	N/A
4	62	120.71	175.78	6	120.71	781.46	N/A	N/A
7	8	38.41	8.87	7	38.41	374.38	N/A	N/A
7	24	122.09	34.86	3	122.09	1291.29	N/A	N/A
7	48	202.54	1631.6	4	202.54	2830.93	N/A	N/A
7	62	263.91	3614.54	4	263.91	9453.89	N/A	N/A

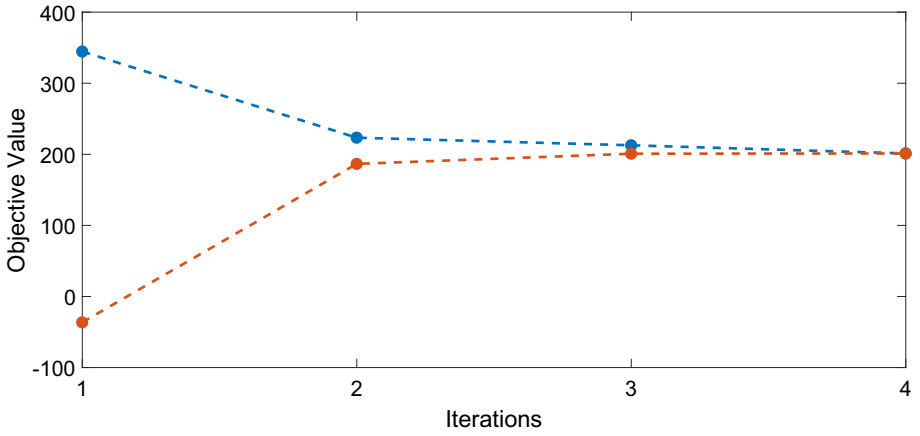

Fig. 4 Progression of the upper and lower bounds during each iteration while solving the 2nd-order version of Problem (22) with 7 rooms and 48 time steps. The blue line depicts the progression of the upper bound and the red line shows that of the lower bound. (Color figure online)

given solution. This may imply that quadratic underapproximating functions and an MIQP coupling step could be more efficient than the current affine implementation. This would also allow for Algorithm 2 to be applicable to a wider class of problems.

It is also worth noting that the majority of the runtime used by Algorithm 2 for the MIQP problem instance is spent in the MILP coupling step. In contrast, the majority of the time spent solving the 4th-order problem is in the MINLP subproblems, and the MILP coupling problems are solved relatively quickly. A breakdown of the runtime per iteration for both of these cases is reported in Tables 8 and 9.

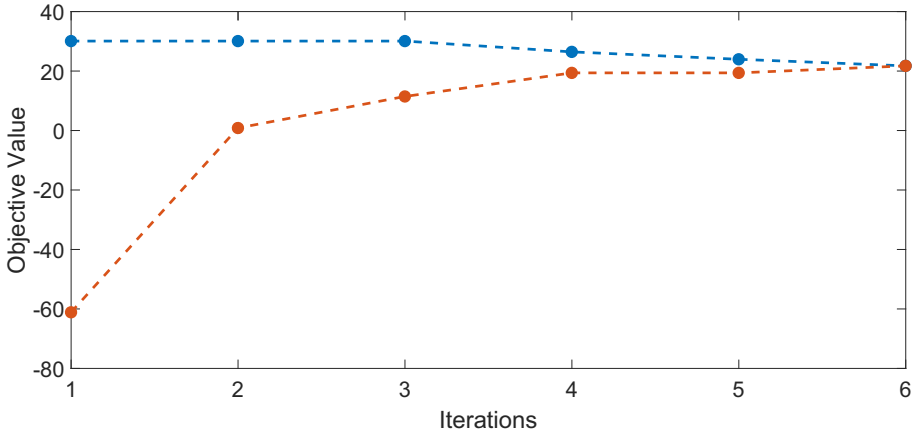

Fig. 5 Progression of the upper and lower bounds during each iteration while solving the 4th-order version of Problem (22) with 4 rooms and 8 time steps. The blue line depicts the progression of the upper bound and the red shows that of the lower bound. (Color figure online)

Table 8 Runtime breakdown of Algorithm 2 applied to the 2nd-order version of Problem 22 with 7 room TCL problem with 48 time steps

	Iter. 1	Iter. 2	Iter. 3	Iter. 4
MINLP time (s)	0.27	0.20	0.20	0.20
MILP time (s)	2.20	86.89	82.62	87.62
Hyperplane time (s)	0.001	0.005	0.002	0.002

Table 9 Runtime breakdown of Algorithm 2 applied to the 4th-order version of Problem 22 with 4 rooms and 8 time steps

	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 6
MINLP time (s)	0.73	0.75	0.67	0.65	0.65	0.66
MILP time (s)	0.03	0.04	0.04	0.04	0.04	0.04
Hyperplane time (s)	0.004	0.006	0.002	0.002	0.002	0.002

4.6 Mixed-Integer Lasso

As a second benchmark example we consider a mixed-integer lasso problem based on Problem (11.1) from [10]. The details of this problem are summarized below for $n + N$ features and M training samples:

$$\min_{w,v} \sum_{j=1}^{M} e^{(-a_j^\top w - b_j^\top z - c_i j v)} + \lambda(||w||_1 + ||z||_1), \tag{23a}$$

subject to:

$$\underline{w} \leq w, v \leq \overline{w}, \tag{23b}$$

$$z \in \mathcal{Z} \subset \mathbb{Z}^N, \tag{23c}$$

where $w \in \mathbb{R}^n$, $v \in \mathbb{R}$. As in [10], the features consist of the tuple (a_j, b_j, c_j) where $a_j \in \mathbb{R}^n$, $b_j \in \mathbb{R}^N$, and $c_j \in \{-1, 1\}$. The regularization parameter λ is chosen as described in [10].

Table 10 Results obtained for Problem (24) using Algorithm 2 and Bonmin

n	M	N	PaDOA (Alg. 2) Obj.	PaDOA (Alg. 2) Time (ms)	PaDOA (Alg. 2) iter.	B-BB Obj.	B-BB Time (ms)
40	100	2	36.42	47,686	392	N/A	4618
100	300	5	105.72	40,925	216	N/A	23,615
40	500	10	284.65	1306	23	284.44	18,723
40	100	10	49.60	742	11	49.53	2055
200	500	10	142.69	20,551	97	N/A	38,050
400	500	10	232.77	352,616	255	N/A	123,071
400	1200	20	361.97	30,473	74	N/A	82,512
600	1800	30	534.71	42,988	66	N/A	228,656
800	2800	40	1029.70	61,997	61	N/A	143,999
1000	2500	50	791.95	66,445	55	N/A	334,536

Assuming that $M \gg N$, Problem (23) can be decomposed into a collection of smaller problems of the form (23) with appropriate consensus constraints. Given the structure of Algorithm 2, it is logical to decompose (23) such that only one integer variable is present per subproblem:

$$\min_{w,v} \sum_{i=1}^{N} \sum_{j=1}^{M_i} e^{(-a_j^\top w_i - b_j z_i - c_j v_i)} + \lambda(||w_i||_1 + ||z_i||_1), \tag{24a}$$

$$\text{subject to } \forall i \in \{1, \ldots, N\}:$$

$$\underline{w}_i \leq w_i, v_i \leq \overline{w}_i, \tag{24b}$$

$$z_i \in \mathcal{Z}_i \subset \mathbb{Z}, \tag{24c}$$

$$w_i = w_k, z_i = z_k, \text{ and } v_i = v_k, \ \forall k \in \{1, \ldots, N\}. \tag{24d}$$

That is, the collection of $M = \sum_{i=1}^{N} M_i$ training samples is decomposed into N (evenly sized) parts, all of which must be in consensus.

4.7 Numerical Results for Mixed-Integer Lasso

To test Algorithm 2 on Problem (24) the results for a variety of problem and partition sizes are presented, with features generated according to [10]. These are displayed in Table 10 with the results of Bonmin used for comparison.

Note that Bonmin with standard settings fails for most of the benchmark problems. Nonetheless, the time spent by Bonmin to attempt a solution is still recorded in Table 10. Overall, Algorithm 2 tends to perform much better with a highly partitioned problem (large N), where each partition contains relatively few decision variables (low n).

Much like the 4th order instance of the TCL problem, the mixed-integer lasso problem requires many iterations until PaDOA converges. Likewise, the upper and lower bounds converge in a similar manner as shown in Fig. 6. That is, the lower bound increases much more quickly than the upper bound decreases.

One interesting difference between the results for the mixed-integer lasso and TCL problems is that the runtime for the TCL problem is much more balanced between the MILP and

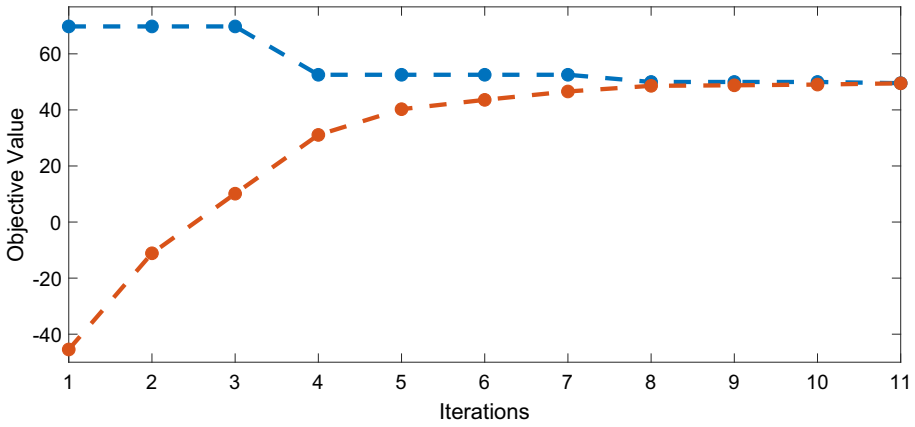

Fig. 6 Progression of the upper and lower bounds during each iteration while solving the Problem (24) with $n = 40$, $M = 100$, and $N = 10$. The blue line depicts the progression of the upper bound and the red shows that of the lower bound. (Color figure online)

Table 11 Runtime breakdown of Algorithm 2 applied to Problem (24) with $n = 40$, $M = 100$, and $N = 10$

Iteration	1	2	3	4	5	6	7	8	9	10	11
MINLP time (ms)	142	15	13	102	13	12	13	14	15	12	17
MILP time (ms)	16	18	15	18	20	19	23	25	26	30	33
Hyp. time (ms)	19	11	11	10	10	11	11	11	11	12	13

MINLP steps, as shown in Table 11. This may be due to the chosen problem partitioning, wherein each MINLP subproblem only requires the solution to a collection of continuous variables and a single integer variable.

5 Conclusions

This paper has introduces a new method (PaDOA) for partially distributing MICP solvers to find ϵ-suboptimal points of the structured MICP (1). PaDOA proceeds by alternating between solving partially decoupled MICPs that comprise m integer variables and large scale MILPs with nN variables. Finite termination conditions for PaDOA are established in Theorem 3. Moreover, we discuss the major theoretical and practical advantages of PaDOA compared to exising extended formulation based OA solvers. In particular, Theorem 2 states that PaDOA terminates after the first iteration, if it is initialized at a global minimizer—an important property that is neither shared by existing OA nor by existing branch-and-bound based methods for MICP.

In Sect. 4, first, second and fourth order mixed integer problems are used to demonstrate the practical performance of PaDOA compared to other state of the art solvers by application to a scheduling problem of thermostatically controlled loads. While the solution and runtime are competitive for each of the case studies considered, the best performance occurs for problems with sparse Hessians and coupling constraints. Furthermore, it is observed that PaDOA is able to return a solution in several cases where the centralized approach fails due to memory

constraints. These results are confirmed with a second benchmark example in Sect. 4.6. It remains to be seen what the specific limitations of this method for distributing MICP solvers are, and a full investigation of PaDOA and a more mature software implementation is still subject to future work.

As shown in Sect. 4, the MIQP implementation of Problem (22) is solved more efficiently than the MILP implementation. Thus, it may be the case that quadratic underapproximating functions and an MIQP coupling step could actually be more efficient for certain problems than the current implementation with affine supporting hyperplanes. This would also have the added benefit of allowing for applicability to a larger problem class. Future work will investigate the use of other under-approximating functions and an extension of Algorithm 2 to non-convex MINLPs. Furthermore, Step 4 requires full constraint information in order to return a feasible solution. This restricts the applicability in terms of fully distributed settings and future work will focus on sidestepping this restriction.

Funding Open Access funding enabled and organized by Projekt DEAL.

Proofs

Proof of Proposition 1

If there is no $x \in X$ with $Ax = b$, both sides of (6) are equal to infinity and the statement of the proposition holds in the extended value sense. Thus, we may assume that the constraints in (6) are feasible. Consequently, Assumption 1 implies that (5) is a convex optimization problem with compact and non-empty feasible set. Moreover, since X is a polytope, all constraints in (5) are linear. It is well-known [6,51] that strong duality holds under these conditions. □

Proof of Lemma 1

If there is no $x \in X$ with $Ax = b$, both sides of (10) are equal to infinity and the statement of the lemma holds in the extended value sense. Thus, we may assume that the equation $Ax = b$ has a solution in X. Next, because we have $z \in \Pi$, our particular construction of Φ implies that

$$\Phi(x, z, \Pi) \geq f^\star(z) + \left[\lambda^\star(z)\right]^\mathsf{T}(x - x^\star(z)) + \underbrace{\left[\mu^\star(z)\right]^\mathsf{T}(z - z)}_{=0} \tag{25}$$

Thus, we have

$$\begin{aligned}\min_{x \in X, Ax = b} \Phi(x, z, \Pi) &= \min_{y \in X, Ay = b} f^\star(z) + \left[\lambda^\star(z)\right]^\mathsf{T}(y - x^\star(z)) \\ &= \min_{x,y} f(x, z) + \left[\lambda^\star(z)\right]^\mathsf{T}(y - x) \quad \text{s.t.} \quad \begin{cases} Ay = b \\ y \in X \end{cases}\end{aligned}$$

$$= \max_{\lambda} \min_{x,y} f(x,z) + \lambda^{\mathsf{T}}(y-x) \quad \text{s.t.} \quad \begin{cases} Ay = b \\ y \in X \end{cases} .$$

Since Assumption 1 holds, we may substitute (6) (see Proposition 1), which yields the equation

$$\forall z \in \Pi, \qquad \min_{x \in X, Ax=b} \Phi(x,z,\Pi) = f^\star(z) .$$

□

Proof of Theorem 1

Notice that if the equation $Ax = b$ has no solution in X, this will be detected immediately by Step 2 of Algorithm 1, which causes termination. Thus, we may assume that all optimization problems are feasible. Now, the main idea of the proof is to show that the cardinality of the set Π is strictly increasing in every iteration, if the algorithm does not terminate. For this aim, we first notice that any solution (x^+, y^+, z^+) of the MILP (12) satisfies the equation

$$\Phi(x^+, z^+, \Pi) = \sum_{i=1}^{N} y_i^+ \tag{26}$$

by construction. Moreover, because we have $Ax^+ = b$, the inequality

$$\min_{x,Ax=b} \Phi(x, z^+, \Pi) \leq \Phi(x^+, z^+, \Pi) \tag{27}$$

holds. If we further assume that the termination criterion is not satisfied, we must have

$$\sum_{i=1}^{N} y_i^+ < U - \epsilon \tag{28}$$

Thus, if we had $z^+ \in \Pi$, then the result of Lemma 1 would imply that

$$f^\star(z^+) \overset{(10)}{=} \min_{x,Ax=b} \Phi(x, z^+, \Pi) \tag{29}$$

as well as $U \leq f(z^+)$, since z^+ has already been added to the collection Π. By substituting all the above relations we would then find that

$$f^\star(z^+) \overset{(29),(27)}{\leq} \Phi(x^+, z^+, \Pi) \overset{(26),(28)}{<} U - \epsilon \leq f(z^+) - \epsilon,$$

which is a contraction. Thus, either our assumption that the algorithm does not terminate or our assumption $z^+ \in \Pi$ must be wrong. In other words, if the algorithm does not terminate in the current step, then the cardinality of the set Π increases by 1 in the next step, because z^+ is added to the collection $\Pi \subseteq Z$. But this is only possible for a finite number of steps, because the set Z contains only a finite number of points. Thus, Algorithm 1 must terminate after a finite number of iterations. □

Proof of Theorem 2

Let $V^\star = \sum_{i=1}^{N} f_i(x_i^\star, z_i^\star)$ denote the optimal value of (1). Because we assume that such an optimal solution exists while Assumption 1 is satisfied, the partially decoupled optimization

problems are all feasible and return piecewise affine lower bounds that satisfy the termination condition (16) with $V_k(z^\star) = V^\star$, i.e., we have

$$V^\star - \epsilon_{\mathrm{L}} \leq \min_{x \in X, \zeta \in Z_k} \Theta_k^\star(x, \zeta) \quad \text{s.t.} \quad Ax = b \tag{30}$$

for all $k \in \{1, \ldots, N\}$. Because the function Θ is by construction an upper bound on Θ_k (for any k), we further have

$$\min_{x \in X, \zeta \in Z_k, Ax=b} \Theta_k^\star(x, \zeta) \leq \min_{x \in X, z \in Z, Ax=b} \Theta(x, z) = \Theta(x^+, z^+),$$

where (x^+, z^+) denotes the solution of the master MILP (20). By substituting the above inequalities we find that

$$V^\star - \epsilon_{\mathrm{L}} \leq \Theta(x^+, z^+) \ .$$

Because we assume that $\epsilon_{\mathrm{L}} \leq \epsilon$, this implies that

$$U - \Theta(x^+, z^+) = V^\star - \Theta(x^+, z^+) \leq \epsilon_{\mathrm{L}} \leq \epsilon \ .$$

Thus, the termination condition is satisfied and Algorithm 2 terminates after the first step. □

Proof of Theorem 3

We may assume that the coupled equality constraint is feasible, as infeasibility would be detected immediately in Step 1 of Algorithm 2. Similar to the proof of Theorem 1, we need to keep track of the integer solutions of the master MILPs. For this aim, we introduce the following "artificial" additional step:

Step 3′): After solving (20), update $\tilde{\Pi} = \tilde{\Pi} \cup \{z^+\}$.

If the set $\tilde{\Pi}$ is initialized with the empty set and if Step 3′ is inserted in Algorithm 2 immediately after Step 3, the iterates of this algorithm remain unaffected. The main idea of the proof is now to show that the cardinality of the set $\tilde{\Pi}$ increases in every iteration of Algorithm 1 under the assumption that the termination criterion is not satisfied. Let us assume that the solution z^+ satisfies $z^+ \in \tilde{\Pi}$ (before $\tilde{\Pi}$ is updated in Step 3′). Then we have

$$U \leq V_k(z^+) \overset{(16)}{\leq} \epsilon_{\mathrm{L}} + \min_{x \in X, \zeta \in Z_k, Ax=b} \Theta_k^\star(x, \zeta),$$

Because Θ is an upper bound on Θ_k, this implies that we also have

$$\min_{x \in X, \zeta \in Z_k, Ax=b} \Theta_k^\star(x, \zeta) \leq \min_{x \in X, z \in Z, Ax=b} \Theta^\star(x, \zeta) = \Phi(x^+, z^+),$$

which yields $U - \Phi(x^+, z^+) \leq \epsilon_{\mathrm{L}} \leq \epsilon$. Thus, either the termination criterion is satisfied or we have $z^+ \notin \tilde{\Pi}$. In the latter case, the cardinality of Π increases by 1 in the current iteration. As this is only possible for a finite number of steps, Algorithm 2 must terminate. □

References

1. Alguacil, N., Motto, A.L., Conejo, A.J.: Transmission expansion planning: a mixed-integer lp approach. IEEE Trans. Power Syst. **18**(3), 1070–1077 (2003)
2. Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M.: CasADi—a software framework for nonlinear optimization and optimal control. Math. Program. Comput. (2018) **(in press)**

3. Andreani, R., Birgin, E.G., Martinez, J.M., Schuverdt, M.L.: On augmented Lagrangian methods with general lower-level constraints. SIAM J. Optim. **18**, 1286–1309 (2007)
4. Arora, J.S.: Introduction to Optimum Design. Elsevier, Amsterdam (2004)
5. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. Numer. Mat. **4**(1), 238–252 (1962)
6. Bertsekas, D.P.: Nonlinear Programming, 2nd edn. Athena Scientific, Belmont (1999)
7. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Wächter, A.: An algorithmic framework for convex mixed integer nonlinear programs. Discrete Optim. **5**(2), 186–204 (2008)
8. Bonami, P., Kilinç, M., Linderoth, J.: Algorithms and software for convex mixed integer nonlinear programs. In: Mixed Integer Nonlinear Programming, vol. 154, pp. 1–39. Springer, New York (2012)
9. Borchers, B., Mitchell, J.E.: An improved branch and bound algorithm for mixed integer nonlinear programs. Comput. Oper. Res. **21**, 359–368 (1994)
10. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2011)
11. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
12. Carrion, M., Arroyo, J.M.: A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. IEEE Trans. Power Syst. **21**(3), 1371–1378 (2006)
13. Conforti, M., Cornuéjols, G., Zambelli, G.: Polyhedral approaches to mixed integer linear programming. In: 50 Years of Integer Programming 1958–2008, pp. 343–385. Springer, New York (2009)
14. Dantzig, G.B., Wolfe, P.: Decomposition principle for linear programs. Oper. Res. **8**(1), 101–111 (1960)
15. Deutscher Wetterdienst. ftp://ftp-cdc.dwd.de/pub/CDC/observations_germany/climate/10_minutes/solar/historical/ (2017)
16. Duran, M.A., Grossmann, I.E.: An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Math. Program. **36**(3), 307–339 (1986)
17. Eckstein, J., Bertsekas, D.P.: On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math. Program. **55**, 293–318 (1992)
18. Eronen, V., Mäkelä, M.M., Westerlund, T.: On the generalization of ECP and OA methods to nonsmooth convex MINLP problems. Optimization **63**(7), 1057–1073 (2014)
19. Everett, H.: Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper. Res. **11**(3), 399–417 (1963)
20. Fletcher, R., Leyffer, S.: Solving mixed integer nonlinear programs by outer approximation. Math. Program. **66**(1–3), 327–349 (1994)
21. Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite element approximations. Comput. Math. Appl. **2**, 17–40 (1976)
22. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences), 1st edn. W. H. Freeman, New York (1979)
23. Geoffrion, A.: Generalized benders decomposition. J. Optim. Theory Appl. **10**, 237–260 (1972)
24. Gupta, O.K., Ravindran, A.: Branch and bound experiments in convex nonlinear integer programming. Manag. Sci. **31**, 1533–1546 (1985)
25. Hijazi, H., Bonami, P., Ouorou, A.: An outer-inner approximation for separable mixed-integer nonlinear programs. INFORMS J. Comput. **26**(1), 31–44 (2014)
26. Houska, B., Frasch, J., Diehl, M.: An augmented Lagrangian based algorithm for distributed Non-Convex optimization. SIAM J. Optim. **26**(2), 1101–1127 (2016)
27. IBM: Using the CPLEX callable library, version 12 (2009)
28. Kesavan, P., Allgor, R.J., Gatzke, E.P., Barton, P.I.: Outer approximation algorithms for separable nonconvex mixed-integer nonlinear programs. Math. Program. **100**(3), 517–535 (2004)
29. Kilinç, M.R., Linderoth, J., Luedtke, J.: Lift-and-project cuts for convex mixed integer nonlinear programs. Math. Program. Comput. **9**(4), 499–526 (2017)
30. Kocuk, B., Dey, S.S., Sun, X.: New formulation and strong MISOCP relaxations for AC optimal transmission switching problem. IEEE Trans. Power Syst. **32**(6), 4161–4170 (2017)
31. Kohlhepp, P., Hagenmeyer, V.: Technical potential of buildings in Germany as flexible power-to-heat storage for smart-grid operation. Energy Technol. **5**(7), 1084–1104 (2017)
32. Kronqvist, J., Bernal, D.E., Grossmann, I.E.: Using regularization and second order information in outer approximation for convex MINLP. Math. Program. **180**(1), 285–310 (2020)
33. Kronqvist, J., Lundell, A., Westerlund, T.: Reformulations for utilizing separability when solving convex MINLP problems. J. Global Optim. **71**(3), 571–592 (2018)
34. Kuindersma, S., Deits, R., Fallon, M., Valenzuela, A., Dai, H., Permenter, F., Koolen, T., Marion, P., Tedrake, R.: Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. Auton. Robot. **40**(3), 429–455 (2016)

35. Leyffer, S.: Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput. Optim. Appl. **18**, 295–309 (2001)
36. Lubin, M., Yamangil, E., Bent, R., Vielma, J.P.: Polyhedral approximation in mixed-integer convex optimization. Math. Program. (2017)
37. Lundell, A., Kronqvist, J., Westerlund, T.: The supporting hyperplane optimizationtoolkit—a polyhedral outer approximation based convex MINLP solver utilizing a single branchingtree approach. Optim. Online (Preprint) (2018)
38. Marriott, K., Stuckey, P.J.: Programming with Constraints: An Introduction. MIT Press, Cambridge (1998)
39. Murray, A., Faulwasser, T., Hagenmeyer, V.: Mixed-integer vs. real-valued formulations of distributed battery scheduling problems. In: 10th Symposium on Control of Power and Energy Systems (CPES 2018) (2018)
40. Murty, K.G., Kabadi, S.N.: Some NP-complete problems in quadratic and nonlinear programming. Math. Program. **39**, 117–129 (1987)
41. Muts, P., Nowak, I., Hendrix, E.M.T.: The decomposition-based outer approximation algorithm for convex mixed-integer nonlinear programming. J. Global Optim. **75**, 1–22 (2020)
42. Muts, P., Nowak, I., Hendrix, E.M.T.: On decomposition and multiobjective-based column and disjunctive cut generation for MINLP. Optim. Eng. 1–30 (2020)
43. Necoara, I., Suykens, J.A.K.: Application of a smoothing technique to decomposition in convex optimization. IEEE Trans. Autom. Control **53**(11), 2674–2679 (2008)
44. Nocedal, J., Wright, S.J.: Sequential Quadratic Programming. Springer, New York (2006)
45. Nowak, I., Breitfeld, N., Hendrix, E.M.T., Njacheun-Njanzoua, G.: Decomposition-based inner-and outer-refinement algorithms for global optimization. J. Global Optim. **72**(2), 305–321 (2018)
46. Nowak, I., Muts, P., Hendrix, E.M.T.: Multi-tree decomposition methods for large-scale mixed integer nonlinear optimization. In: Large Scale Optimization in Supply Chains and Smart Manufacturing, pp. 27–58. Springer (2019)
47. Gurobi Optimization: Gurobi optimizer reference manual (2009)
48. Powell, M.J.D.: A method for nonlinear constraints in minimization problems. In: Fletcher, R. (ed.) Optimization. Academic Press, Cambridge (1969)
49. Quesada, I., Grossmann, I.E.: An LP/NLP based branch-and-bound algorithm for convex MINLP optimization problems. Comput. Chem. Eng. **16**, 937–947 (1992)
50. Ravemark, D.E., Rippin, D.W.T.: Optimal design of a multi-product batch plant. Comput. Chem. Eng. **22**, 177–183 (1998)
51. Rockafellar, R.T.: Convex analysis (1970)
52. Rückmann, J., Shapiro, A.: Augmented Lagrangians in semi-infinite programming. Math. Program. Ser. B **116**, 499–512 (2009)
53. Sawaya, N.: Reformulations, relaxations and cutting planes for generalized disjunctive programming. Ph.D. thesis, Carnegie Mellon University (2006)
54. Shapiro, A., Sun, J.: Some properties of the augmented Lagrangian in cone constrained optimization. Math. Oper. Res. **29**(3), 479–491 (2004)
55. Takapoui, R., Möhle, N., Boyd, S., Bemporad, A.: A simple effective heuristic for embedded mixed-integer quadratic programming. Int. J. Control **86**, 1–11 (2016)
56. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**(2), 225–249 (2005)
57. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. J. Optim. Theory Appl. **109**(3), 475–494 (2001)
58. Vielma, J.P., Dunning, I., Huchette, J., Lubin, M.: Extended formulations in mixed integer conic quadratic programming. Math. Program. Comput. **95**, 1–50 (2016)
59. Westerlund, T., Pettersson, F.: A cutting plane method for solving convex MINLP problems. Comput. Chem. Eng. **19**, 131–136 (1995)
60. Wright, S.: Coordinate descent algorithms. Math. Program. Ser. B **151**(1), 3–34 (2015)
61. Zhang, W., Kalsi, K., Fuller, J., Elizondo, M., Chassin, D.: Aggregate model for heterogeneous thermostatically controlled loads with demand response. In: 2012 IEEE Power and Energy Society General Meeting, pp. 1–8 (2012)