Optimization and Engineering (2023) 24:801–830
https://doi.org/10.1007/s11081-021-09707-y

RESEARCH ARTICLE

Comparison of MINLP formulations for global superstructure optimization

Jannik Burre[1] · **Dominik Bongartz**[1] · **Alexander Mitsos**[1]

Received: 11 August 2021 / Revised: 27 November 2021 / Accepted: 21 December 2021 /
Published online: 13 January 2022

Abstract

Superstructure optimization is a powerful but computationally demanding task that can be used to select the optimal structure among many alternatives within a single optimization. In chemical engineering, such problems naturally arise in process design, where different process alternatives need to be considered simultaneously to minimize a specific objective function (e.g., production costs or global warming impact). Conventionally, superstructure optimization problems are either formulated with the Big-M or the Convex Hull reformulation approach. However, for problems containing nonconvex functions, it is not clear whether these yield the most computationally efficient formulations. We therefore compare the conventional problem formulations with less common ones (using equilibrium constraints, step functions, or multiplications of binary and continuous variables to model disjunctions) using three case studies. First, a minimalist superstructure optimization problem is used to derive conjectures about their computational performance. These conjectures are then further investigated by two more complex literature benchmarks. Our analysis shows that the less common approaches tend to result in a smaller problem size, while keeping relaxations comparably tight—despite the introduction of additional nonconvexities. For the considered case studies, we demonstrate that all reformulation approaches can further benefit from eliminating optimization variables by a reduced-space formulation. For superstructure optimization problems containing nonconvex functions, we therefore encourage to also consider problem formulations that introduce additional nonconvexities but reduce the number of optimization variables.

Keywords Global optimization · Superstructure · Big-M · Convex Hull · McCormick · MINLP

✉ Alexander Mitsos
amitsos@alum.mit.edu

[1] Process Systems Engineering (AVT.SVT), RWTH Aachen University, Forckenbeckstraße 51, 52074 Aachen, Germany

1 Introduction

A major problem arising in process synthesis is the simultaneous selection of process equipment and their optimization regarding design and operation. Superstructure optimization represents a suitable but computationally demanding approach to solve such problems. Typical formulations include discrete variables. However, in some cases the formulations are purely continuous (e.g., blending problem for fuels (Singh et al. 2000), process water networks (Ahmetović and Grossmann 2011)). Superstructures can be represented in several ways resulting in different problem formulations for which different solution algorithms exist (Grossmann 2002; Mencarelli et al. 2020). A common and intuitive modeling approach is their formulation as generalized disjunctive programming (GDP) problems consisting of algebraic constraints, disjunctions, Boolean variables, and logic propositions (Raman and Grossmann 1994). These GDP problems can be solved directly with dedicated solution algorithms (e.g., logic-based Outer Approximation (Lee and Grossmann 2000) or GDP branch and bound (Türkay and Grossmann 1996b)) that aim to exploit the disjunctions effectively (e.g., by directly branching on them or reducing the problem size by considering only the active disjuncts). Alternatively, they can be reformulated to mixed-integer nonlinear programming (MINLP) problems (Grossmann and Trespalacios 2013), in which integer variables replace Boolean variables and algebraic equations model the logic propositions.

Conventional reformulation approaches are the Big-M (Nemhauser and Wolsey 1988) and Convex Hull (Lee and Grossmann 2000) method. The reformulation of GDP to MINLP problems enables the use of powerful commercial MINLP solvers and thus the possibility to solve GDP problems containing nonconvex functions (Lee and Grossmann 2003; Ruiz and Grossmann 2010). In contrast, existing solvers dedicated to GDP problems are restricted to GDP problems containing only convex functions. It is also possible to avoid introducing Boolean or binary variables for modeling the disjunctions by using complementarity constraints resulting in mathematical programs with equilibrium constraints (MPEC) (Luo et al. 1996). This results in a special type of nonlinear programming (NLP) problem with only a few optimization variables. The NLP problem can be either solved directly or again reformulated (e.g., by using the Plus Function (Chen and Mangasarian 1996)), which often involves smoothing techniques. In summary, a number of modeling approaches and solution algorithms has been developed over the past decades. Their comparison has mainly been limited to the conventional approaches and problems with only linear or convex constraints representing the process models. For problems containing nonconvex functions, which require global optimization techniques, a comparative assessment of all approaches mentioned above is missing so far.

For global optimization, several techniques for improving computational tractability have been proposed. As superstructure optimization problems often have many variables and constraints, factorable reduced-space (RS) formulations can be beneficial (for an overview of these formulations, see Bongartz (2020), Chapter 3). In such RS formulations, equality constraints are used to eliminate optimization variables from the problem, while ensuring that the optimization problem remains factorable, i.e., all functions can still be expressed as compositions of simple so-called intrinsic functions (i.e., functions for which convex and concave relaxations are known). Compared to the more conventional (equation-oriented) full-space (FS) formulations of the problem, the RS formulations have fewer optimization variables and constraints. In the branch-and-bound (BaB) algorithms that are used for global optimization of nonconvex problems, RS formulations reduce the dimensionality of the space that needs to be partitioned via branching, and they reduce the size of the subproblems for computing lower and upper bounds on the optimal objective value. Depending on the method used for constructing relaxations, they may however result in weaker relaxations (see Bongartz (2020), p. 64). Factorable RS formulations have been used successfully for flowsheet optimization problems (Byrne and Bogle 2000; Bongartz and Mitsos 2017, 2019), parameter estimation problems involving ordinary differential equations (Mitsos et al. 2009), and problems with machine learning models embedded (Schweidtmann and Mitsos 2018; Schweidtmann et al. 2021). Its application to superstructure optimization problems has not been investigated yet. Even more variables and constraints can be eliminated from the problem by dropping the requirement that the functions remain factorable, and instead allowing the use of implicit functions defined by the constraints. Barton and co-workers have presented methods for handling such implicit functions in BaB algorithms (Scott et al. 2011; Stuber et al. 2014; Wechsung et al. 2015). However, these are beyond the scope of this work as corresponding implementations are not readily available yet.

In the present work, we compare different problem formulations for superstructure optimization problems comprising nonconvex functions. Within this comparison, we investigate whether the optimization can benefit from a RS formulation.

In Sect. 2, we introduce a simple and a more complex illustrative example problem. For the simple example problem, the conventional and unconventional problem formulations are developed in Sect. 3. The comparison of all problem formulations for the two example problems are presented in Sect. 4. In Sect. 5, the key results are confirmed by an optimization problem with a different structure (i.e., piecewise-defined cost function instead of unit selection), which can be modeled using the same reformulation approaches. Sect. 6 concludes our findings.

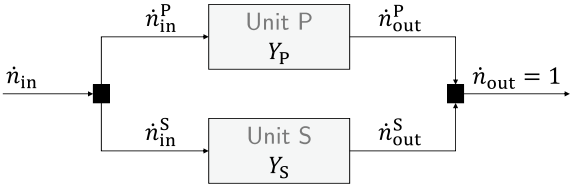

Fig. 1 Schematic of the simple example problem with the choice of producing a desired amount of a product, $\dot{n}_{out} = 1$, either via Unit P (Y_P = True, Y_S = False) or Unit S (Y_P = False, Y_S = True)

2 Problem definition

To introduce the different problem formulations in Sect. 3, we use a simple illustrative example problem (Sect. 2.1). This example problem and a more complex one with multiple disjunctions (Sect. 2.2) are then used to analyze each problem formulation in greater detail in Sect. 4.

2.1 Simple illustrative example problem

In this simple example problem, we consider only two mutually exclusive options (either Unit P or Unit S) to produce a desired amount of a product, $\dot{n}_{out} = 1$ (considered as a parameter). This illustrative example problem is motivated by our work on power-to-X (Burre et al. 2020, 2021; Roh et al. 2020), in which decisions need to be taken on how to provide raw materials (e.g., hydrogen and carbon dioxide) and process them toward the final product (e.g., electricity-based fuels).

Each option comes with operating and investment costs, both of which are dependent on molar flow rates and unit-specific cost parameters (Table 1). Operating costs C_{op} are quadratically dependent on the molar flow rate passing through the chosen unit ($C_{op}^{j} = (\dot{n}_{in}^{j})^2 e_j$). Investment costs C_{inv} are dependent on the maximal rating given as the global feed flow rate into the superstructure, $\dot{n}_{in}$, and a constant cost parameter ($C_{inv}^{j} = C_j + \dot{n}_{in}^{0.6}$). The dependency on $\dot{n}_{in}$ (in addition to $\dot{n}_{in}^{j}$ for operating costs) puts more emphasis on the nonconvexity regarding multiple variables in typical process synthesis problems. This concave term can also be moved out of the disjunction directly into the objective function. Our simple numerical experimentation showed that this does not have any effect on the optimization. Also the consideration of a constant conversion parameter for each process unit does not have an influence on the overall results. To keep the illustrative example problem simple, we do not consider such a parameter.

The objective is to find the process with the lowest total cost C by solving the following optimization problem formulated as a GDP problem:

Table 1 Cost parameters for the simple (GDP1) and the multiple-disjunction (GDP2) example problem. C_j are fixed investment costs and e_j are specific operating costs

Parameter	Unit P	Unit S	Unit F1	Unit F2
e_j	7	3	0.1	0.3
C_j	4	7	0.5	0.4

$$
\begin{aligned}
\min \quad & C = C_{\text{op}} + C_{\text{inv}} \\
\text{s.t.} \quad & \dot{n}_{\text{in}} = \dot{n}^{\text{P}}_{\text{in}} + \dot{n}^{\text{S}}_{\text{in}} \\
& \dot{n}_{\text{out}} = \dot{n}^{\text{P}}_{\text{out}} + \dot{n}^{\text{S}}_{\text{out}} \\
& \begin{bmatrix} Y_{\text{P}} \\ \dot{n}^{\text{S}}_{\text{in}} = \dot{n}^{\text{S}}_{\text{out}} = 0 \\ \dot{n}^{\text{P}}_{\text{out}} = \dot{n}^{\text{P}}_{\text{in}} \\ C_{\text{op}} = (\dot{n}^{\text{P}}_{\text{in}})^2 e_{\text{P}} \\ C_{\text{inv}} = C_{\text{P}} + \dot{n}^{0.6}_{\text{in}} \end{bmatrix} \veebar \begin{bmatrix} Y_{\text{S}} \\ \dot{n}^{\text{P}}_{\text{in}} = \dot{n}^{\text{P}}_{\text{out}} = 0 \\ \dot{n}^{\text{S}}_{\text{out}} = \dot{n}^{\text{S}}_{\text{in}} \\ C_{\text{op}} = (\dot{n}^{\text{S}}_{\text{in}})^2 e_{\text{S}} \\ C_{\text{inv}} = C_{\text{S}} + \dot{n}^{0.6}_{\text{in}} \end{bmatrix} \\
& Y_{\text{P}} \veebar Y_{\text{S}} \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \geq 0 \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \in \mathbb{R} \\
& Y_{\text{P}}, Y_{\text{S}} \in \{\text{True}, \text{False}\}
\end{aligned}
\tag{GDP1}
$$

There are alternative ways for modeling the system, e.g., by eliminating the global (i.e., independent from disjunctions) equality constraints and modifying the disjunctive constraints correspondingly. As Problem (GDP1) is the most direct representation of the superstructure shown in Fig. 1, the use of these global constraints is considered throughout this work. Additionally, we explicitly restrict all illustrative example problems to the exclusive choice between units (denoted by the logic exclusive "or"-operator $\veebar$) to keep it as simple as possible.

2.2 Multiple-disjunction example problem

As the flowsheet structure of Problem (GDP1) is minimalist, we analyze a more complex example problem with two disjunctions taken from (Grossmann and Trespalacios 2013), where the outlet of Unit S needs to be further processed by either Unit F1 or F2 (Fig.2). To make the choice for one of these units economically viable and the case study interesting, we introduce a cost factor of 0.1 for Unit F1 and F2.

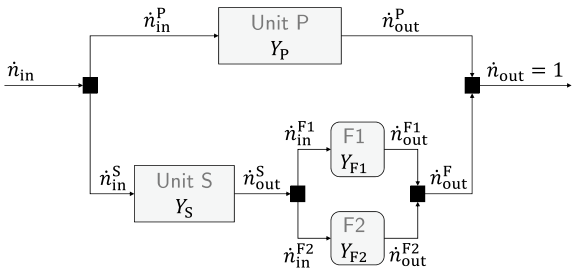

Fig. 2 Schematic of the flowsheet structure with two disjunctions and the choice of producing a desired amount of a product, $\dot{n}_{\text{out}} = 1$, either via Unit P ($Y_{\text{P}} = 1$, $Y_{\text{S}} = 0$) or Unit S ($Y_{\text{P}} = 0$, $Y_{\text{S}} = 1$). If Unit S is chosen, the product needs to be further processed by either Unit F1 ($Y_{\text{F1}} = 1$, $Y_{\text{F2}} = 0$) or F2 ($Y_{\text{F1}} = 0$, $Y_{\text{F2}} = 1$)

The more complex flowsheet structure results in Problem (P2) (ESI Sect. 3), which is further reformulated to Problem (GDP2) to remove the nested structure within the disjunction and derive corresponding problem formulations as described in Sect. 3. These are summarized in ESI Sect. 3.

$$
\begin{aligned}
\min \quad & C = C_{\text{op}} + C_{\text{inv}} \\
\text{s.t.} \quad & \dot{n}_{\text{in}} = \dot{n}_{\text{in}}^{\text{P}} + \dot{n}_{\text{in}}^{\text{S}} \\
& \dot{n}_{\text{out}}^{\text{S}} = \dot{n}_{\text{in}}^{\text{F1}} + \dot{n}_{\text{in}}^{\text{F2}} \\
& \dot{n}_{\text{out}}^{\text{F}} = \dot{n}_{\text{out}}^{\text{F1}} + \dot{n}_{\text{out}}^{\text{F2}} \\
& \dot{n}_{\text{out}} = \dot{n}_{\text{out}}^{\text{P}} + \dot{n}_{\text{out}}^{\text{F}} \\
& \begin{bmatrix} Y_{\text{P}} \\ \dot{n}_{\text{in}}^{\text{S}} = \dot{n}_{\text{out}}^{\text{S}} = 0 \\ \dot{n}_{\text{out}}^{\text{P}} = \dot{n}_{\text{in}}^{\text{P}} \\ C_{\text{op}} = (\dot{n}_{\text{in}}^{\text{P}})^2 e_{\text{P}} \\ C_{\text{inv}} = C_{\text{P}} + \dot{n}_{\text{in}}^{0.6} \end{bmatrix} \vee \begin{bmatrix} Y_{\text{S}} \\ \dot{n}_{\text{in}}^{\text{P}} = \dot{n}_{\text{out}}^{\text{P}} = 0 \\ \dot{n}_{\text{out}}^{\text{S}} = \dot{n}_{\text{in}}^{\text{S}} \\ C_{\text{op}} = (\dot{n}_{\text{in}}^{\text{S}})^2 e_{\text{S}} + C_{\text{op}}^{\text{F}} \\ C_{\text{inv}} = C_{\text{S}} + \dot{n}_{\text{in}}^{0.6} + C_{\text{inv}}^{\text{F}} \end{bmatrix} \\
& \begin{bmatrix} Y_{\text{F1}} \\ \dot{n}_{\text{in}}^{\text{F2}} = \dot{n}_{\text{out}}^{\text{F2}} = 0 \\ \dot{n}_{\text{out}}^{\text{F1}} = \dot{n}_{\text{in}}^{\text{F1}} \\ C_{\text{op}}^{\text{F}} = (\dot{n}_{\text{in}}^{\text{F1}})^2 e_{\text{F1}} \\ C_{\text{inv}}^{\text{F}} = C_{\text{F1}} + 0.1 \dot{n}_{\text{in}}^{0.6} \end{bmatrix} \vee \begin{bmatrix} Y_{\text{F2}} \\ \dot{n}_{\text{in}}^{\text{F1}} = \dot{n}_{\text{out}}^{\text{F1}} = 0 \\ \dot{n}_{\text{out}}^{\text{F2}} = \dot{n}_{\text{in}}^{\text{F2}} \\ C_{\text{op}}^{\text{F}} = (\dot{n}_{\text{in}}^{\text{F2}})^2 e_{\text{F2}} \\ C_{\text{inv}}^{\text{F}} = C_{\text{F2}} + 0.1 \dot{n}_{\text{in}}^{0.6} \end{bmatrix} \\
& \vee \begin{bmatrix} Y_{\text{notF}} \\ \dot{n}_{\text{in}}^{\text{F1}} = \dot{n}_{\text{in}}^{\text{F2}} = \dot{n}_{\text{out}}^{\text{F1}} = \dot{n}_{\text{out}}^{\text{F2}} = 0 \\ C_{\text{op}}^{\text{F}} = 0 \\ C_{\text{inv}}^{\text{F}} = 0 \end{bmatrix} \\
& Y_{\text{P}} \veebar Y_{\text{S}} \\
& Y_{\text{F1}} \veebar Y_{\text{F2}} \veebar Y_{\text{notF}} \\
& Y_{\text{S}} \Leftrightarrow Y_{\text{F1}} \veebar Y_{\text{F2}} \\
& C_{\text{op}}, C_{\text{inv}}, C_{\text{op}}^{\text{F}}, C_{\text{inv}}^{\text{F}}, \dot{n}_{\text{in}}, \dot{n}_{\text{in}}^{\text{P}}, \dot{n}_{\text{in}}^{\text{S}}, \dot{n}_{\text{in}}^{\text{F1}}, \dot{n}_{\text{in}}^{\text{F2}}, \dot{n}_{\text{out}}^{\text{P}}, \dot{n}_{\text{out}}^{\text{S}}, \dot{n}_{\text{out}}^{\text{F1}}, \dot{n}_{\text{out}}^{\text{F2}}, \dot{n}_{\text{out}}^{\text{F}} \geq 0 \\
& C_{\text{op}}, C_{\text{inv}}, C_{\text{op}}^{\text{F}}, C_{\text{inv}}^{\text{F}}, \dot{n}_{\text{in}}, \dot{n}_{\text{in}}^{\text{P}}, \dot{n}_{\text{in}}^{\text{S}}, \dot{n}_{\text{in}}^{\text{F1}}, \dot{n}_{\text{in}}^{\text{F2}}, \dot{n}_{\text{out}}^{\text{P}}, \dot{n}_{\text{out}}^{\text{S}}, \dot{n}_{\text{out}}^{\text{F1}}, \dot{n}_{\text{out}}^{\text{F2}}, \dot{n}_{\text{out}}^{\text{F}} \in \mathbb{R} \\
& Y_{\text{P}}, Y_{\text{S}}, Y_{\text{F1}}, Y_{\text{F2}}, Y_{\text{notF}} \in \{\text{True}, \text{False}\}
\end{aligned}
\tag{GDP2}
$$

3 Problem formulations

In this section, the most common problem formulations for superstructure optimization and less common ones are presented and applied to the simple illustrative example problem (GDP1). One of the most conventional methods to reformulate GDP into MINLP problems are the *Big-M* (Sect. 3.1, (Nemhauser and Wolsey 1988)) and the *Convex Hull* method (Sect. 3.2, (Lee and Grossmann 2000)). Big-M

and Convex Hull transform Boolean variables Y_j into binary variables y_j to express the relationship between and within disjunctions. In contrast, a nonsmooth reformulation approach can be used to prevent the use of any additional variables completely by introducing complementarity constraints. The resulting *MPEC*s can be either solved directly (Sect. 3.3, (Baumrucker et al. 2008)) or again reformulated using the *Plus Function* (Sect. 3.4, (Chen and Mangasarian 1996)). Another alternative problem formulation, hereafter called *Direct MINLP*, utilizes binary variables directly within the equality constraints of the process model to capture the existence or nonexistence of a unit or process stream within a disjunction (Sect. 3.5). As we also consider RS formulations for all reformulation approaches, Sect. 3.6 illustrates such a RS formulation exemplarily for the Direct MINLP reformulation approach. In addition to the aforementioned reformulation approaches, superstructures representing problems with piecewise-defined functions can also be handled by using step functions (cf. Sect. 5, (Wechsung and Barton 2013)).

3.1 Big-M

The Big-M method introduces the parameter M (a big number potentially individual for each constraint of disjunct j), which is used to make the constraints of unchosen choices ($y_j = 0$) redundant. Several methods exist in literature that determine the optimal value for M and thus tighten relaxations (e.g., (Trespalacios and Grossmann 2015)). To maintain a manageable complexity, we do not consider such improved Big-M methods. Instead, we choose M to be equal to the global upper bound of the variable that is bounded by the corresponding constraint. Besides binary variables for each disjunct (the transformed Boolean variables), no additional variables need to be introduced. However, a poor choice for M can result in weak relaxations. Applied to Problem (GDP1), the Big-M method results in Problem (BM1):

$$
\begin{aligned}
\min \quad & C = C_{\mathrm{op}} + C_{\mathrm{inv}} \\
\text{s.t.} \quad & \dot{n}_{\mathrm{in}} = \dot{n}^{\mathrm{P}}_{\mathrm{in}} + \dot{n}^{\mathrm{S}}_{\mathrm{in}} \\
& \dot{n}_{\mathrm{out}} = \dot{n}^{\mathrm{P}}_{\mathrm{out}} + \dot{n}^{\mathrm{S}}_{\mathrm{out}} \\
& 0 - M(1 - y_{\mathrm{P}}) \le \dot{n}^{\mathrm{S}}_{k} \le 0 + M(1 - y_{\mathrm{P}}) && k \in \{\mathrm{in}, \mathrm{out}\} \\
& 0 - M(1 - y_{\mathrm{S}}) \le \dot{n}^{\mathrm{P}}_{k} \le 0 + M(1 - y_{\mathrm{S}}) && k \in \{\mathrm{in}, \mathrm{out}\} \\
& \dot{n}^{j}_{\mathrm{in}} - M(1 - y_j) \le \dot{n}^{j}_{\mathrm{out}} \le \dot{n}^{j}_{\mathrm{in}} + M(1 - y_j) && j \in \{\mathrm{P}, \mathrm{S}\} \\
& (\dot{n}^{j}_{\mathrm{in}})^2 e_j - M(1 - y_j) \le C_{\mathrm{op}} \le (\dot{n}^{j}_{\mathrm{in}})^2 e_j + M(1 - y_j) && j \in \{\mathrm{P}, \mathrm{S}\} \\
& C_j + \dot{n}^{0.6}_{\mathrm{in}} - M(1 - y_j) \le C_{\mathrm{inv}} \le C_j + \dot{n}^{0.6}_{\mathrm{in}} + M(1 - y_j) && j \in \{\mathrm{P}, \mathrm{S}\} \\
& y_{\mathrm{P}} + y_{\mathrm{S}} = 1 \\
& C_{\mathrm{op}}, C_{\mathrm{inv}}, \dot{n}_{\mathrm{in}}, \dot{n}^{\mathrm{P}}_{\mathrm{in}}, \dot{n}^{\mathrm{S}}_{\mathrm{in}}, \dot{n}^{\mathrm{P}}_{\mathrm{out}}, \dot{n}^{\mathrm{S}}_{\mathrm{out}} \ge 0 \\
& C_{\mathrm{op}}, C_{\mathrm{inv}}, \dot{n}_{\mathrm{in}}, \dot{n}^{\mathrm{P}}_{\mathrm{in}}, \dot{n}^{\mathrm{S}}_{\mathrm{in}}, \dot{n}^{\mathrm{P}}_{\mathrm{out}}, \dot{n}^{\mathrm{S}}_{\mathrm{out}} \in \mathbb{R} \\
& y_{\mathrm{P}}, y_{\mathrm{S}} \in \{0, 1\}
\end{aligned}
\tag{BM1}
$$

3.2 Convex Hull

For the Convex Hull method, a new (disaggregated) variable, ν_j, needs to be introduced for each variable that is affected by a disjunction and for each choice within the respective disjunction. The bounds on these disaggregated variables can be either chosen to be the global variable bounds or tightened based on the disjunct. In our illustrative example problems, we use global variable bounds (i.e., $0 \leq \dot{n} \leq 1$, $0 \leq C \leq 20$). Each constraint $r_j(\nu_j) \leq 0$ of disjunct j is expressed by the closure of the perspective function (Ceria and Soares 1999):

$$y_j r_j\left(\frac{\nu_j}{y_j}\right) \leq 0 \tag{1}$$

To avoid singularities for $y_j = 0$, Lee and Grossmann (2000) proposed a modification of the original perspective function, which was again modified by Sawaya (2006) to improve numerical performance and accuracy:

$$((1-\epsilon)y_j + \epsilon) r_j\left(\frac{\nu_j}{(1-\epsilon)y_j + \epsilon}\right) \leq 0 \tag{2}$$

For the illustrative example problems, parameter ϵ only needs to be nonzero as the disaggregated variables of unchosen units ($\nu_j = 0$) make Constraint (2) being fulfilled independently from the value of ϵ. In general, ϵ needs to be chosen sufficiently small to maintain a high accuracy. As Constraints (2) represents the commonly used type of perspective function for global optimization, we use it for all Convex Hull formulations within this study where necessary. Other modifications exist and are summarized in Furman et al. (2020). In this study, they are not considered.

The introduction of disaggregated variables increases problem size but generally yields tighter relaxations compared to the Big-M formulation, even if the most suitable Big-M parameter is utilized (Grossmann and Lee 2003). The conversion of Problem (GDP1) with the Convex Hull method results in Problem (CH1):

$$
\begin{aligned}
\min \quad & C = C_{\text{op}} + C_{\text{inv}} \\
\text{s.t.} \quad & \dot{n}_{\text{in}} = \dot{n}^{\text{P}}_{\text{in}} + \dot{n}^{\text{S}}_{\text{in}} \\
& \dot{n}_{\text{out}} = \dot{n}^{\text{P}}_{\text{out}} + \dot{n}^{\text{S}}_{\text{out}} \\
& \dot{n}^{j}_{\text{in}} = \dot{n}^{j}_{\text{in,P}} + \dot{n}^{j}_{\text{in,S}} && j \in \{\text{P}, \text{S}\} \\
& \dot{n}^{j}_{\text{out}} = \dot{n}^{j}_{\text{out,P}} + \dot{n}^{j}_{\text{out,S}} && j \in \{\text{P}, \text{S}\} \\
& C_{\text{op}} = C^{\text{P}}_{\text{op}} + C^{\text{S}}_{\text{op}} \\
& C_{\text{inv}} = C^{\text{P}}_{\text{inv}} + C^{\text{S}}_{\text{inv}} \\
& \dot{n}^{\text{S}}_{k,\text{P}}, \dot{n}^{\text{P}}_{k,\text{S}} \leq 0 && k \in \{\text{in}, \text{out}\} \\
& C^{j}_{\text{op}} \geq \left(\frac{\dot{n}^{j}_{\text{in},j}}{(1-\epsilon)y_j + \epsilon} \right)^2 e_j((1-\epsilon)y_j + \epsilon) && j \in \{\text{P}, \text{S}\} \\
& C^{j}_{\text{inv}} \geq (C_j + \dot{n}^{0.6}_{\text{in}})y_j && j \in \{\text{P}, \text{S}\} \\
& y_{\text{P}} + y_{\text{S}} = 1 \\
& 0 \leq \dot{n}^{\text{S}}_{k,\text{P}} \leq 1 y_{\text{P}} && k \in \{\text{in}, \text{out}\} \\
& 0 \leq \dot{n}^{\text{P}}_{k,\text{S}} \leq 1 y_{\text{S}} && k \in \{\text{in}, \text{out}\} \\
& 0 \leq \dot{n}^{j}_{k,j} \leq 1 y_j && k \in \{\text{in}, \text{out}\} \quad j \in \{\text{P}, \text{S}\} \\
& 0 \leq C^{j}_{m} \leq 20 y_j && m \in \{\text{op}, \text{inv}\} \quad j \in \{\text{P}, \text{S}\} \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \geq 0 \\
& C_{\text{op}}, C_{\text{inv}}, C^{\text{P}}_{\text{op}}, C^{\text{S}}_{\text{op}}, C^{\text{P}}_{\text{inv}}, C^{\text{S}}_{\text{inv}} \in \mathbb{R} \\
& \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in,P}}, \dot{n}^{\text{S}}_{\text{in,P}}, \dot{n}^{\text{P}}_{\text{in,S}}, \dot{n}^{\text{S}}_{\text{in,S}} \in \mathbb{R} \\
& \dot{n}^{\text{P}}_{\text{out,P}}, \dot{n}^{\text{S}}_{\text{out,P}}, \dot{n}^{\text{P}}_{\text{out,S}}, \dot{n}^{\text{S}}_{\text{out,S}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \in \mathbb{R} \\
& y_{\text{P}}, y_{\text{S}} \in \{0, 1\}
\end{aligned}
\tag{CH1}
$$

3.3 MPEC

In order to prevent the use of discrete variables and thus prevent solving a MINLP problem, superstructure optimization problems can be formulated as MPECs. This type of problem formulation introduces complementarity constraints to model the discrete choices in a process superstructure.

The MPEC problem formulation can be derived from Problem (GDP1) by considering all global constraints (to represent the part of the flowsheet that is not affected by disjunctions) and a subset of disjunctive constraints (to represent the part of the flowsheet that is affected by disjunctions). The subset of disjunctive constraints is chosen in such a way that (a) only those model equations of a disjunction are considered that correspond to the part of the flowsheet where $Y_j = \text{True}$ ($\dot{n}^{\text{P}}_{\text{out}} = \dot{n}^{\text{P}}_{\text{in}}$, $C^{\text{P}}_{\text{op}} = (\dot{n}^{\text{P}}_{\text{in}})^2 e_{\text{P}}$, and $C^{\text{P}}_{\text{inv}} = C_{\text{P}} + \dot{n}^{0.6}_{\text{in}}$ for the choice $Y_{\text{P}} = \text{True}$; $\dot{n}^{\text{S}}_{\text{out}} = \dot{n}^{\text{S}}_{\text{in}}$,

$C^{\mathrm{S}}_{\mathrm{op}} = (\dot{n}^{\mathrm{S}}_{\mathrm{in}})^2 e_{\mathrm{S}}$, and $C^{\mathrm{S}}_{\mathrm{inv}} = C_{\mathrm{S}} + \dot{n}^{0.6}_{\mathrm{in}}$ for the choice $Y_{\mathrm{S}} = \mathrm{True}$) and (b) zero values can be realized if the choice j within the disjunction is not active. However, this is not given for all types of disjunctive problems: In the simple illustrative example problem (GDP1), C^{j}_{inv} contains constant investment costs C_j for unit j and therefore need to be modified by a step function that is dependent on the material stream $\dot{n}^{j}_{\mathrm{in}}$ passing through unit j. Such a step function introduces nonconvexities additionally to those that are inherently part of the modeled system into the problem. If tailored relaxations are implemented for such a step function (Wechsung and Barton 2013), the optimization may however not be necessarily affected negatively. For all problem formulations in this work, the tanh-function is used as a smoothed step function resulting in an error below the feasibility tolerance. By adding up the cost terms C^{j}_{op} and C^{j}_{inv} for each unit j, operating costs C_{op} and investment costs C_{inv} are then retrieved, respectively. The relationship between disjunctions and choices within each disjunction is represented by complementarity constraints instead of binary variables. For Problem (MPEC1), the molar flows passing each unit are multiplied by each other and set to zero. Therefore, either $\dot{n}^{\mathrm{P}}_{\mathrm{in}}$ or $\dot{n}^{\mathrm{S}}_{\mathrm{in}}$ need to become zero if the other one is nonzero. Due to the absence of any discrete variables, global NLP solvers can be used to find the global optimum. This can however be challenging. Although a considerably smaller problem size can be achieved in comparison to the other approaches, the introduced complementarity constraints can result in the problem violating constraint qualifications and hence cause problems for the local solvers used for upper bounding. Often, a regularization parameter (a small number μ) need to be added to make the constraint qualifications hold again. The solution of the original problem is then obtained by sequentially reducing μ to zero (Scholtes 2001). Such a regularization is however not required for the problems analyzed in this work, as the global solvers used herein do in practice not depend heavily on the performance of the NLP solver for upper bounding, for which the constraint qualifications need to hold. To improve performance, BARON detects complementary constraints automatically and treats them accordingly. In MAiNGO, there is no special algorithm to detect and treat complementarity constraints implemented yet.

The MPEC formulation of Problem (GDP1) using complementarity constraints results in Problem (MPEC1), in which parameter P (a big number) is used to approximate the step function more accurately.

$$
\begin{aligned}
\min \quad & C = C_{\text{op}} + C_{\text{inv}} \\
\text{s.t.} \quad & \dot{n}_{\text{in}} = \dot{n}^{\text{P}}_{\text{in}} + \dot{n}^{\text{S}}_{\text{in}} \\
& \dot{n}^{j}_{\text{out}} = \dot{n}^{j}_{\text{in}} \qquad\qquad j \in \{\text{P}, \text{S}\} \\
& \dot{n}_{\text{out}} = \dot{n}^{\text{P}}_{\text{out}} + \dot{n}^{\text{S}}_{\text{out}} \\
& C_{\text{op}} = \sum_{j \in J} (\dot{n}^{j}_{\text{in}})^2 e_j \\
& C_{\text{inv}} = \sum_{j \in J} \left[\tanh(P \dot{n}^{j}_{\text{in}})(C_j + \dot{n}^{0.6}_{\text{in}}) \right] \\
& 0 = \dot{n}^{\text{P}}_{\text{in}} \dot{n}^{\text{S}}_{\text{in}} \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \geq 0 \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}^{\text{P}}_{\text{in}}, \dot{n}^{\text{S}}_{\text{in}}, \dot{n}^{\text{P}}_{\text{out}}, \dot{n}^{\text{S}}_{\text{out}} \in \mathbb{R}
\end{aligned}
\tag{MPEC1}
$$

3.4 Plus Function

An alternative representation of the complementarity constraint ($0 = \dot{n}^{\text{P}}_{\text{in}} \dot{n}^{\text{S}}_{\text{in}}$) in Problem (MPEC1) can be achieved by using the Plus Function (Chen and Mangasarian 1996):

$$
0 = \dot{n}^{\text{P}}_{\text{in}} - \max(0, \dot{n}^{\text{P}}_{\text{in}} - \dot{n}^{\text{S}}_{\text{in}}). \tag{3}
$$

This function is commonly used for modeling the nonsmooth behavior of flash units for vapor-liquid(-liquid) equilibrium calculations with vanishing phases (Gopal and Biegler 1999; Sahlodin et al. 2016), for which it has shown a promising computational performance. We refer to the resulting problem formulation as Problem (PLUS1) (ESI Section 2).

3.5 Direct MINLP

A problem formulation for superstructure optimization that is only barely used is the direct multiplication of binary variables with the continuous variables being present in the disjunctions. Similarly to the aforementioned problem formulations, it can be derived from GDP problem (GDP1). Boolean variables Y_j of the GDP problem transform to binary variables y_j and are (in contrast to the Big-M and Convex Hull method) multiplied with the continuous model variables ($\dot{n}^{j}_{\text{in}} = y_j \dot{n}_{\text{in}}$ and $C_{\text{inv}} = \sum_{j \in J} y_j (C_j + \dot{n}^{0.6}_{\text{in}})$) to represent the existence or nonexistence of option j within the process flowsheet. Instead of introducing additional inequality constraints modeling the disjunctions, the Direct MINLP formulation incorporates the disjunctions directly into the algebraic equality constraints of the model. Doing this, potential redundant global constraints may need to be disregarded ($\dot{n}_{\text{in}} = \dot{n}^{\text{P}}_{\text{in}} + \dot{n}^{\text{S}}_{\text{in}}$) and the variables in the objective function for each choice in the disjunction added up ($C_{\text{op}} = \sum_{j \in J} (\dot{n}^{j}_{\text{in}})^2 e_j$ and $C_{\text{inv}} = \sum_{j \in J} y_j (C_j + \dot{n}^{0.6}_{\text{in}})$). The model formulation results in Problem (MINLP1):

$$
\begin{aligned}
\min \quad & C = C_{\text{op}} + C_{\text{inv}} \\
\text{s.t.} \quad & C_{\text{op}} = \sum_{j \in J} (\dot{n}_{\text{in}}^{j})^2 e_j \\
& C_{\text{inv}} = \sum_{j \in J} y_j (C_j + \dot{n}_{\text{in}}^{0.6}) \\
& \dot{n}_{\text{in}}^{j} = y_j \dot{n}_{\text{in}} && j \in \{\text{P}, \text{S}\} \\
& \dot{n}_{\text{out}}^{j} = \dot{n}_{\text{in}}^{j} && j \in \{\text{P}, \text{S}\} \\
& \dot{n}_{\text{out}} = \dot{n}_{\text{out}}^{\text{P}} + \dot{n}_{\text{out}}^{\text{S}} \\
& y_{\text{P}} + y_{\text{S}} = 1 \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}_{\text{in}}^{\text{P}}, \dot{n}_{\text{in}}^{\text{S}}, \dot{n}_{\text{out}}^{\text{P}}, \dot{n}_{\text{out}}^{\text{S}} \geq 0 \\
& C_{\text{op}}, C_{\text{inv}}, \dot{n}_{\text{in}}, \dot{n}_{\text{in}}^{\text{P}}, \dot{n}_{\text{in}}^{\text{S}}, \dot{n}_{\text{out}}^{\text{P}}, \dot{n}_{\text{out}}^{\text{S}} \in \mathbb{R} \\
& y_j \in \{0, 1\} && j \in \{\text{P}, \text{S}\}
\end{aligned}
\tag{MINLP1}
$$

In contrast to the conventional problem formulations, the Direct MINLP formulation introduces nonlinearities by the multiplication of the binary variables y_j with expressions of the continuous variable $\dot{n}_{\text{in}}$ as part of the algebraic equality constraints in the model. If the remaining model is linear or convex, the reformulation using the Big-M (cf. Sect. 3.1) or Convex Hull (cf. Sect. 3.2) method results in a mixed-integer linear programming (MILP) or convex MINLP problem, respectively, which can be efficiently solved with general-purpose solvers. Thus, for MILP and convex MINLP problems, the conventional reformulation approaches are typically superior to alternative approaches. If the remaining model is nonconvex anyway, e.g., in more detailed process engineering, where nonconvexities are usually inherently part of the (mechanistic) process model, it is not clear whether it is beneficial to reformulate it with the conventional approaches, as it does not result in a MILP or convex MINLP problem (Huster et al. 2020). Keeping bilinear terms according to the Direct MINLP formulation can result in smaller subproblems, which can be directly given to MINLP solvers.

From a modeling perspective, we may also view Problem (MINLP1) instead of Problem (GDP1) as a starting formulation, which may again be reformulated using the Big-M or Convex Hull approach. This results in slightly larger problems compared to Problem (BM1) and (CH1) and was therefore found to be less computationally efficient than the direct reformulation of Problem (GDP1).

3.6 Reduced-space formulation

The problem formulations stated in the preceding sections are given in their conventional FS formulation. As we also consider RS formulations (i.e., eliminating optimization variables and constraints), we exemplarily introduce the RS formulation of Problem (MINLP1) in such a way that variables y that depend on other model variables x are written as factorable functions $\tilde{y}(x)$. By doing so, the objective function

becomes an explicit function of the degrees of freedom x only. This can be done for all problem formulations. The RS formulation for Problem (MINLP1) is

$$\begin{aligned} \min_{y_\mathrm{P}, \dot{n}_\mathrm{in}} \quad & \tilde{C}(y_\mathrm{P}, \dot{n}_\mathrm{in}) \\ \text{s.t.} \quad & \dot{n}_\mathrm{out} - \dot{\tilde{n}}^\mathrm{P}_\mathrm{out}(y_\mathrm{P}, \dot{n}_\mathrm{in}) - \dot{\tilde{n}}^\mathrm{S}_\mathrm{out}(y_\mathrm{P}, \dot{n}_\mathrm{in}) = 0, \\ & \dot{n}_\mathrm{in} \geq 0 \\ & \dot{n}_\mathrm{in} \in \mathbb{R} \\ & y_\mathrm{P} \in \{0, 1\}, \end{aligned} \tag{MINLP1 RS}$$

which has only two variables $(y_\mathrm{P}, \dot{n}_\mathrm{in})$ and contains the following functions:

$$\begin{aligned} \tilde{C}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= \tilde{C}_\mathrm{op}(y_\mathrm{P}, \dot{n}_\mathrm{in}) + \tilde{C}_\mathrm{inv}(y_\mathrm{P}, \dot{n}_\mathrm{in}) \\ \tilde{C}_\mathrm{op}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= (\dot{\tilde{n}}^\mathrm{P}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}))^2 \cdot \mathrm{e}_\mathrm{P} + (\dot{\tilde{n}}^\mathrm{S}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}))^2 \cdot \mathrm{e}_\mathrm{S} \\ \tilde{C}_\mathrm{inv}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= y_\mathrm{P} \cdot (C_\mathrm{P} + \dot{n}_\mathrm{in}^{0.6}) + \tilde{y}_\mathrm{S}(y_\mathrm{P}) \cdot (C_\mathrm{S} + \dot{n}_\mathrm{in}^{0.6}) \\ \dot{\tilde{n}}^\mathrm{P}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= y_\mathrm{P} \cdot \dot{n}_\mathrm{in} \\ \dot{\tilde{n}}^\mathrm{S}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= \tilde{y}_\mathrm{S}(y_\mathrm{P}) \cdot \dot{n}_\mathrm{in} \\ \tilde{y}_\mathrm{S}(y_\mathrm{P}) &:= 1 - y_\mathrm{P} \\ \dot{\tilde{n}}^\mathrm{P}_\mathrm{out}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= \dot{\tilde{n}}^\mathrm{P}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}) \\ \dot{\tilde{n}}^\mathrm{S}_\mathrm{out}(y_\mathrm{P}, \dot{n}_\mathrm{in}) &:= \dot{\tilde{n}}^\mathrm{S}_\mathrm{in}(y_\mathrm{P}, \dot{n}_\mathrm{in}). \end{aligned}$$

Such a RS formulation with intermediate variables computed as a function of other variables can easily be implemented in procedural modeling environments, e.g., via the C++ or Python APIs of MAiNGO, which then uses the MC++ library (Chachuat et al. 2015) to obtain relaxations of these functions. The elimination of variables in such procedural modeling environments corresponds to the construction of a sequence of mathematical operations, for which a relaxation and their subgradients are constructed. This procedure can be considered as propagating the relaxations through the algorithm (Mitsos et al. 2009).

4 Results for the illustrative example problems

As we focus on superstructure optimization problems containing nonconvex functions, we do not consider dedicated solvers for GDP problems but rather reformulate the optimization problem either into a MINLP or a (nonsmooth) NLP problem (cf. Sect. 3). The resulting nonconvex optimization problems generally exhibit multiple suboptimal minima, such that a global solver needs to be used. We use our open-source deterministic global solver MAiNGO v0.5.0.2 (Bongartz et al. 2018), which employs a standard BaB algorithm with several bound tightening techniques and the multivariate McCormick method (McCormick 1976; Tsoukalas and Mitsos 2014) implemented in MC++ (Chachuat et al. 2015) to obtain relaxations. For a comparison with state-of-the-art deterministic global solvers, we perform each optimization also with the commercial solver BARON v19.3.24 (Kılınç and Sahinidis 2017) in

the modeling system GAMS 27.0.0 using the automated generation of GAMS files by MAiNGO. In contrast to MAiNGO, BARON uses the auxiliary variable method (AVM) (Smith and Pantelides 1997; Tawarmalani and Sahinidis 2002), which replaces each nonlinear term by an auxiliary variable (AV) and a constraint. For such constraints, known relaxations are constructed. This method is also used by other state-of-the-art deterministic global solvers such as ANTIGONE (Misener and Floudas 2014) and SCIP (Achterberg 2009), the analysis of which is however beyond the scope of this work. All calculations are conducted on an Intel® Core™i3-6100 CPU with 3.7 GHz running Windows 10 and no other applications. For both global solvers, default settings are selected. The optimality and feasibility tolerance is 10^{-3} and 10^{-6}, respectively. Each optimization has been performed 100 times to reduce the variations in solution time for small problems caused by system background processes. For all problems, the arithmetic mean solution time is reported in this study.

4.1 Simple illustrative example problem

For the simple illustrative example problem (GDP1), the problem size and numerical results of each problem formulation are summarized in Table 2. FS formulations treat all model variables as optimization variables, whereas RS formulations use model equality constraints to eliminate as many optimization variables as possible. One of the remaining equality constraints in the RS formulations makes sure that the sum of streams $\dot{n}^{\mathrm{P}}_{\mathrm{out}}$ and $\dot{n}^{\mathrm{S}}_{\mathrm{out}}$ (i.e., the exiting stream $\dot{n}_{\mathrm{out}}$) equals 1. The second remaining equality constraint in Problem (MPEC1) and (PLUS1) is the complementarity constraint modeling the logic proposition of Problem (GDP1). As the complementarity equality constraint does not eliminate a degree of freedom, the problem still has one degree of freedom despite having the same number of variables and equality constraints. The remaining inequality constraints in the RS formulations for Problem (BM1) and (CH1)

Table 2 Problem size and numerical results for the problem formulations of Problem (GDP1) presented in Sect. 3. The optimization has been executed 100 times, of which the arithmetic mean value is shown. The optimal value is 11

	Big-M		Convex Hull		MPEC		Plus Function		Direct MINLP	
	(BM1)		(CH1)		(MPEC1)		(PLUS1)		(MINLP1)	
	FS	RS	FS	RS	FS	RS	FS	RS	FS	RS
Number of										
Continuous variables	7	2	19	2	7	2	7	2	7	1
Discrete variables	2	1	2	1	0	0	0	0	2	1
Equality constraints	3	1	9	1	7	2	7	2	8	1
Inequality constraints	20	4	34	4	0	0	0	0	0	0
BaB nodes	3	3	3	3	1	5	1	5	1	3
Lower bound of root node	0	5.75	5	4.5	11	2.12	11	2.12	11	8. 5
CPU time per BaB node / s	0.058	0.053	0.094	0.068	0.094	0.005	0.04	0.005	0.097	0.022
Solution time / s	0.174	0.16	0.282	0.204	0.094	0.027	0.04	0.026	0.097	0.066

result from the remaining disjunctive optimization variables. In the FS formulation, the Convex Hull formulation leads to much larger problems than its alternatives, whereas in the RS formulation, the corresponding problem is the same size as that resulting from the Big-M formulation. The unconventional reformulation approaches however still lead to smaller problem sizes, which gets more pronounced for more complex problems (cf. Sect. 4.2 and 5).

In overall, the RS formulations in MAiNGO yield significantly lower overall solution times than corresponding FS formulations and are the lowest for the unconventional reformulation approaches. The higher number of BaB nodes and the lower lower bound (LB) in the root node for Problem (BM1) compared to those of Problem (MPEC1), (PLUS1), and (MINLP1) in the FS formulation indicate weaker relaxations despite the additional nonconvexities introduced by the unconventional approaches. For Problem (CH1), the optimization in FS suffers from the high number of optimization variables (21, cf. Table 2). The RS formulation accelerates the optimization by about one third of the solution time in the FS formulation (0.20 s vs. 0.28 s) given its smaller problem size. Compared to the alternative problem formulations, the optimization still takes longer.

The solution time for Problem (MPEC1) in FS is similar to that of Problem (MINLP1) in FS. In RS, solution time reductions are again possible despite weaker relaxations. The same findings also apply to Problem (MINLP1). The Plus Function formulation of the complementarity constraints in Problem (PLUS1) seems to reduce computational effort even further.

Compared to the commercial global solver BARON, optimization with MAiNGO for the unconventional problem formulations (MPEC1) and (MINLP1) is comparably fast for the FS formulations and even faster for the RS formulations, whereas BARON can handle the conventional problem formulations (BM1) and (CH1) better (especially for the FS formulations). This tendency was expected, as MAiNGO explicitly exploits the benefits of a smaller problem size in the lower bounding problem by using McCormick relaxations (Tsoukalas and Mitsos 2014) opposed to the AVM used in BARON.

If linear instead of the nonlinear cost functions are considered, the Big-M and Convex Hull method results in MILP problems (cf. Problem (BM1lin) and (CHlin), respectively, ESI Sect. 1), which can be solved using CPLEX for both the FS and RS formulation. In contrast, the alternative reformulation approaches (yielding Problem (MPEC1lin), (PLUS1lin), and (MINLP1lin), ESI Section 1) still result in MINLP problems with nonconvex functions, which need to be solved using a global solver. However, the overall results for these three problems considering linear cost functions (cf. ESI Table S1 and ESI Fig. S1) do not differ from those with nonconvex cost functions. This indicates that the nonconvexity of model equations seem to have only minor influence on the optimization of this simple illustrative example problem.

4.2 Multiple-disjunction example problem

Table 3 summarizes the problem size and numerical results of each problem formulation for Problem (GDP2) (each problem formulation is given in ESI Section 3). The newly introduced inequality constraint in the RS formulation of Problem

(MINLP2) makes sure that the binary variable y_{F2} can not become negative. Beyond the equality constraint for the leaving stream $\dot{n}_{out}$, Problem (BM2) and (CH2) require an additional equality constraint for the RS formulation to ensure that exactly one choice is being made for the second disjunction (choice for either F1, F2, or none). Problem (MINLP2) does not need a binary for this third choice (neither F1 nor F2) as the algebraic equations for representing the disjunctions differ from those that result directly from reformulating Problem (GDP2) using the Big-M and Convex Hull approach (cf. ESI Problem (BM2) and (CH2)). As a result, no additional equality constraint is required. Similarly to the simple illustrative example problem (cf. Sect. 4.1), the RS formulations of Problem (MPEC2), (PLUS2), and (MINLP2) lead to the smallest problems.

As for the simple illustrative example problem (cf. Sect. 4.1), the RS formulations in MAiNGO always yield lower overall solution times than corresponding FS formulations. However, the differences between the conventional and unconventional reformulation approaches are for the RS formulations not as pronounced as in the previous example problem. In contrast, they are still well-marked for the FS formulations. In the FS formulation, Problem (MPEC2) and (PLUS2) perform similarly due to their same size and similar relaxation tightness (cf. Table 3). Despite Problem (MINLP2) having four additional binary variables, its solution time is similar to that of the aforementioned problem formulations. Problem (BM2) and (MINLP2) benefit from a significantly smaller problem size compared to Problem (CH2), such that their time consumed per BaB node is also significantly smaller. The complementarity-constrainted problems (MPEC2) and (PLUS2) do not exploit their small problem size so effectively. The prominent advantage of the Convex Hull method—achieving tight relaxations—does not seem to be significant here: The relaxations of the complementarity-constrained problem formulations (MPEC2) and (PLUS2) in FS seem to be even tighter (higher LB in root node) and the optimization requires fewer BaB nodes than Problem (CH2) (cf. Table 3). For Problem (BM2), MAiNGO seems to have considerable problems in the FS formulation. The solution time exceeds 30 minutes, which is most likely caused by the simplistic handling of integer variables in MAiNGO. There are no sophisticated heuristics for generating integer-feasible points implemented yet, which can result in poor performance. In RS formulations, this becomes less likely because of the much lower number of possible branches. However, relaxations generally tend to become weaker in the RS formulation, which is confirmed by the lower LB in the root node for RS compared to the FS formulations. The effect of the reduction in problem size on overall solution time is however bigger than that of the relaxation tightness, which is impressively demonstrated for Problem (CH2). All in all, the much simpler and potentially more intuitive problem formulations (MPEC2), (PLUS2), and (MINLP2) seem to benefit from their comparably small problem sizes both in the FS and RS formulation while maintaining comparatively tight relaxations. They always yield solution times as low as or lower than that of the more complex formulations (BM2) and (CH2).

The comparison of the results obtained by MAiNGO with those obtained by BARON confirms the findings from the simple illustrative example problem (cf. Sect. 4.1), yet to a smaller extent: BARON performs better than MAiNGO for FS formulations with only few nonconvex terms (Problem (BM2) and (CH2)), whereas

it performs slightly worse for problems in the RS formulation with a higher number of nonconvex terms (Problem (MPEC2) and (MINLP2)).

4.3 Selective branching for FS formulations

As an alternative to reducing the problem size on the modeling level (cf. Sect. 1), the problem size can also be reduced on the algorithm level in the means of *selective branching* (Epperly and Pistikopoulos 1997; Dür and Horst 1997; Ben-Tal et al. 1994). The crucial difference between operating in a *reduced space* on the modeling level (RS formulation in MAiNGO) and on the algorithm level (selective branching) is that selective branching only reduces the dimensionality of the space that needs to be partitioned via branching, whereas the RS formulation in MAiNGO additionally reduces the dimension of subproblems to be solved in both the lower and upper bounding problem. In selective branching, the same (significant) reductions in the number of BaB nodes required to solve the optimization problem and the overall solution time have been reported (Epperly and Pistikopoulos 1997). This does however apply only to problems with a specific structure and if constraint propagation is used to remain a high convergence order (Kannan and Barton 2017).

Irrespective of the approach for reformulating the GDP, the RS formulation in MAiNGO always results in lower solution times than the corresponding FS formulation for both the simple and more complex flowsheet structure (Fig. 3 and 4 , respectively). In contrast, the LB in the root node is always lower and the number of BaB nodes required for global optimality is always larger for the RS formulation than for the FS formulation or they are equal, which indicates weaker relaxations for the RS formulations caused by the propagation of relaxations through the algorithm.

For comparison and to isolate the effect of a smaller problem size from that of the branching behavior, we perform selective branching on the RS optimization variables in the FS formulations. This selective branching should result in similar BaB trees between the FS and RS formulations in terms of selection of branching variables and points, while it still exploits the potentially tighter relaxations of the FS

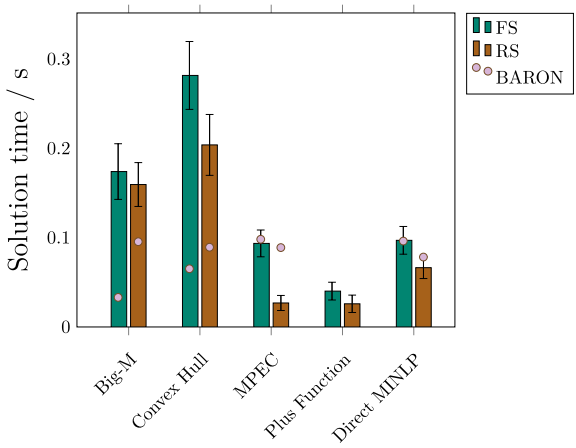

Fig. 3 Solution time for Problem (GDP1) using the problem formulations presented in Sect. 3. As BARON can not handle *max*-functions, there are no results for the Plus Function formulation. The error bars represent the standard deviation from the arithmetic mean value of the solution time from 100 optimization runs using MAiNGO

Table 3 Problem size and numerical results for the problem formulations of Problem (GDP2) presented in Sect. 3. The optimization has been executed 100 times, of which the arithmetic mean value is shown The optimal value is 11.7

	Big-M		Convex Hull		MPEC		Plus Function		Direct MINLP	
	(BM2)		(CH2)		(MPEC2)		(PLUS2)		(MINLP2)	
	FS	RS	FS	RS	FS	RS	FS	RS	FS	RS
Number of										
Continuous variables	14	3	44	3	12	3	12	3	12	1
Discrete variables	5	2	5	2	0	0	0	0	4	2
Equality constraints	8	2	20	2	11	2	11	2	14	1
Inequality constraints	54	12	86	29	0	0	0	0	0	1
BaB nodes	706,798	13	3	7	1	25	1	7	7	7
Lower bound of root node	0	5.8	11.36	5.75	11.7	1.52	11.7	1.52	6.19	3.65
CPU time per BaB node / s	0.003	0.009	0.238	0.017	0.195	0.003	0.135	0. 009	0.035	0.012
Solution time / s	2,193	0.112	0.714	0.119	0.195	0.079	0.135	0. 064	0.248	0.087

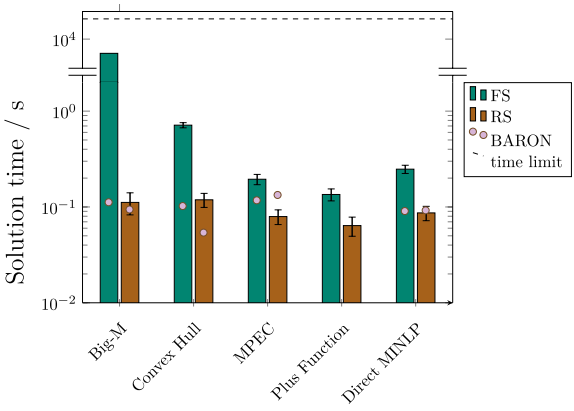

Fig. 4 Solution time for Problem (GDP2) using the problem formulations presented in Sect. 3. As BARON can not handle *max*-functions, there are no results for the Plus Function formulation. The error bars represent the standard deviation from the arithmetic mean value of the solution time from 100 optimization runs using MAiNGO

formulations. This investigation is performed for the more complex flowsheet structure (Problem (GDP2)) as the differences in solution times for the FS and RS formulations are most pronounced for this problem and their branching variables differ from each other.

Except for Problem (BM2), which seems to have an anomaly in the FS formulation, the results in Table 4 and Fig. 5 show that the solution time remains about the same if selective branching is applied to the FS formulations. This shows that the reduction in solution time is exclusively due to the smaller size of the subproblems. For Problem (MINLP2), the selective branching results in a lower number of BaB nodes compared to its FS formulation without selective branching. This has however only a negligible effect on the overall solution time compared to the solution times for RS formulations as the time consumed per BaB node for the RS formulation is considerably smaller.

Table 4 Problem size and numerical results for the FS formulation of Problem (GDP2) if selective branching is applied. The optimization has been executed 100 times, of which the arithmetic mean value is shown. The optimal value is 11.7. For the FS formulation of Problem (BM2), the maximum solution time of 86,400 s has been reached

	Big-M (BM2)	Convex Hull (CH2)	MPEC (MPEC2)	Plus Function (PLUS2)	Direct MINLP (MINLP2)
Number of					
Continuous variables	14	44	12	12	12
Discrete variables	5	5	0	0	4
Equality constraints	8	20	11	11	14
Inequality constraints	54	86	0	0	0
BaB nodes	8,138,810	3	1	1	5
Lower bound of root node	0	11.36	11.7	11.7	6.19
CPU time per BaB node / s	0.011	0.241	0.197	0.136	0.048
Solution time / s	86,400	0.723	0.197	0.136	0.24

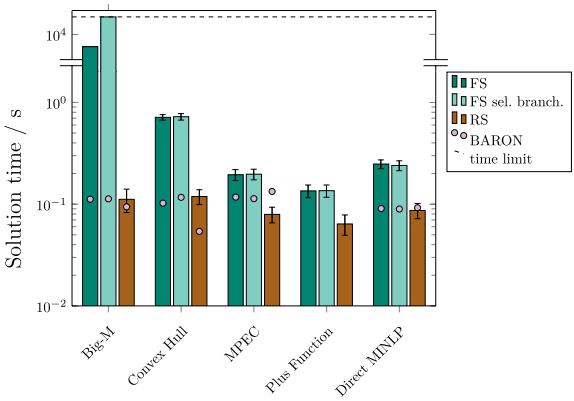

Fig. 5 Solution time for the FS formulation of Problem (GDP2) if selective branching is applied compared to both the FS formulation without selective branching and the RS formulation. As BARON can not handle *max*-functions, there are no results for the Plus Function formulation. The error bars represent the standard deviation from the arithmetic mean value of the solution time from 100 optimization runs using MAiNGO

Selective branching in BARON via the specification of branching priorities has also only negligible influence on the solution time and the number of BaB nodes.

5 Application to problems with piecewise-defined functions

A commonly used case study for testing problem formulations with piecewise-defined functions is the solution of a heat exchanger network design problem, which can be formulated as a superstructure optimization problem. This is a special case of superstructure optimization problems as the alternative units can be described as one single unit with a piecewise-defined function, in this case, of the heat exchanger area A_i. Thus, the disjuncts do not represent different units but rather different regions within the piecewise-defined cost function of each unit (heat exchanger). The case study is taken from Türkay and Grossmann (1996a) and is depicted in Fig. 6 with parameters given in Table 5. The corresponding formulation of Problem (HEXGDP) can be found in ESI Sect. 4.1.

For the RS formulations, model equations had to be rearranged (e.g., for the elimination of T_1, by a partial fraction decomposition) in order to tighten relaxations and prevent singularities. Since in this example the alternative units can also be described as a single unit containing a piecewise-defined (potentially discontinuous) function, we introduce a new problem formulation (Problem (HEXStepF), ESI Sect. 4), hereafter called *Step Formulation*. This formulation involves step functions, for which McCormick relaxations can be constructed (Wechsung and Barton 2013). The piecewise-defined function

$$\phi(x) = \begin{cases} \phi_1(x) & \text{if } x \leq x_1 \\ \phi_2(x) & \text{otherwise} \end{cases} \tag{4}$$

is reformulated as follows:

$$\phi(x) = \pi(x - x_1)\phi_2(x) + [1 - \pi(x - x_1)]\phi_1(x), \tag{5}$$

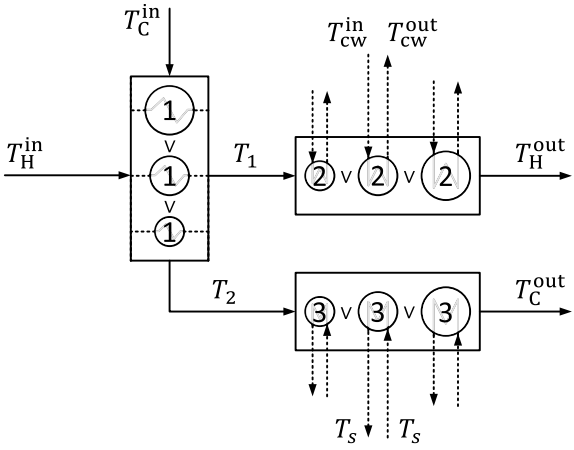

Fig. 6 Heat exchanger network based on the work of Türkay and Grossmann (1996a) with the choice between three different sizes for each heat exchanger 1–3

where

$$\pi(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

The problem size and numerical results for each problem formulation (see ESI Sect. 4 for each formulation) are summarized in Table 6. For both global solvers, default settings are used.

The key findings from the previous case studies are confirmed by this discontinuous optimization problem. The RS formulations always result in shorter overall optimization times mainly resulting from the reduced problem size. The negative effect of the elimination of optimization variables on the tightness of the root node relaxation is particularly pronounced for this case study. However, for Problem (HEXBM), (HEXCH), and (HEXMINLP), this has only a minor negative influence on the number of BaB nodes required. For the other problems, the number of BaB nodes does even decrease much likely due to a more directed branching. Despite the lowest LB in the root node of Problem (HEXMINLP), it requires the fewest BaB nodes and thus outperforms the Big-M and Convex Hull formulation for the RS formulation in this case study.

Reformulation approaches avoiding the introduction of binary variables and thus leading to NLP problems (Problem (HEXMPEC), (HEXPLUSF, and (HEXStepF)) do not benefit in the FS formulation from their smaller problem size. Their subproblems are solved quickly, but a high number of BaB nodes is required despite yielding the tightest relaxations in the root node among all formulations. Moving to RS formulations reduces solution time considerably as significantly fewer BaB nodes are required. Since relaxations can never be tighter in RS than corresponding FS formulations, the reduced number of BaB nodes can mainly be attributed to a more directed branching. The combination of a more directed branching with a small problem size resulting from RS formulations for the reformulation approaches introducing additional nonconvexities yield the

Table 5 Parameters for the heat exchanger network design problem taken from Türkay and Grossmann (1996a)

Stream	FCP_i / kW K^{-1}	T^{in} / K	T^{out} / K	C / \$ kW^{-1} yr^{-1}
H	10.0	500	340	N/A
C	7.5	350	560	N/A
cw	N/A	300	320	20
s	N/A	600	600	80

Heat exchanger	Overall heat transfer / kW m^{-2} K^{-1}
1	1.5
2	0.5
3	1

Heat exchanger	Area / m^2	Costs / \$yr^{-1}
1,2, and 3	$0 < A \leq 10$	$2750A^{0.6} + 3000$
	$10 \leq A \leq 25$	$1500A^{0.6} + 15000$
	$25 \leq A \leq 50$	$600A^{0.6} + 46500$

Table 6 Problem size and numerical results for Problem (HEXGDP) with problems formulations presented in Sect. 3 and the Step Formulation. The optimization has been executed 100 times, of which the arithmetic mean value is shown. The optimal value is 114,385. No results are given for the RS (AVM/McCormick hybrid) Step Formulation, as it does not contain repeated nonlinear terms that could be replaced by AVs

	Big-M (HEXBM)			Convex Hull (HEXCH)			MPEC (HEXMPEC)		
	FS	RS	RS (hybrid)	FS	RS	RS (hybrid)	FS	RS	RS (hybrid)
Number of									
Continuous variables	14	9	9	50	9	9	14	9	9
Discrete variables	9	6	6	9	6	6	0	0	0
Equality constraints	7	2	2	19	2	2	10	5	5
Inequality constraints	72	58	58	72	25	25	0	0	0
BaB nodes	47	138	32,000,000	63	87	47	5,728	593	111
Lower bound of root node	500	− 221,505	− 120,625	15,952	− 230,505	− 130,549	19,534	− 230,505	− 230,505
CPU time per BaB node / s	0.02	0.007	0.003	0.024	0.006	0.012	0.002	0.001	0.001
Solution time / s	0.958	0.931	86,400	1.513	0.535	0.48	8.896	0.643	0.156

	Plus Function (HEXPLUSF)			Direct MINLP (HEXMINLP)			Step Formulation (HEXStepF)	
	FS	RS	RS (hybrid)	FS	RS	RS (hybrid)	FS	RS
Number of								
Continuous variables	14	9	9	14	9	9	8	3
Discrete variables	0	0	0	9	6	6	0	0

Table 6 (continued)

	Plus Function (HEXPLUSF)			Direct MINLP (HEXMINLP)			Step Formulation (HEXStepF)	
	FS	RS	RS (hybrid)	FS	RS	RS (hybrid)	FS	RS
Equality constraints	10	5	5	10	2	2	7	2
Inequality constraints	0	0	0	0	3	3	0	0
BaB nodes	1,838	1,721	271	19	61	27	11,774	187
Lower bound of root node	19,534	− 230,505	− 230,505	2,388	− 300,115	− 83,131	1,404	− 189,277
CPU time per BaB node / s	0.002	0.001	0.002	0.12	0.006	0.011	0.001	0.001
Solution time / s	3.51	2.024	0.55	2.281	0.35	0.287	14.643	0.224

lowest computational effort. Also the overall trend, BARON performing better than MAiNGO for the conventional reformulations approaches and MAiNGO generally performing better than BARON for the unconventional approaches in the RS formulations (except for the MINLP formulation), is confirmed by this heat exchanger network design problem.

We have demonstrated that RS formulations have the big advantage of a smaller problem size and a more directed branching, while they suffer from weaker relaxations than FS formulations. To combine the advantage of a small problem size from the RS formulations with the relaxation tightness of the FS formulations, we can selectively consider AVs for repeated nonlinear terms, thus achieving a hybrid (Najman et al. 2021) between McCormick relaxations and the AVM. Such a hybrid can be easily considered by MAiNGO using its automated identification and replacement of repeated nonlinear terms with AVs. In general, it is much more likely for RS than FS formulations to contain these repeated nonlinear terms (see Bongartz (2020), p.64ff.). In fact, for the heat exchanger network design problem, only the RS formulations contain repeated nonlinear terms. As illustrated in Table 6 and Fig. 7, the selected addition of AVs is beneficial especially for the complementarity-constrained problems (HEXMPEC) and (HEXPLUSF) despite an unchanged LB in the root node. However, the number of BaB nodes reduces considerably while the CPU time per BaB node remains about the same. In contrast, for the other problem formulations (except Problem (HEXBM)) the subproblems seem to become more difficult to solve, such that the benefit of the tighter relaxations resulting from adding selected AVs is almost outweighed. Problem (HEXStepF) does not exhibit repeated nonlinear terms in both the FS and RS formulations. The analysis shows: The consideration of selected AVs within the RS formulations has the potential to improve computational effort even further.

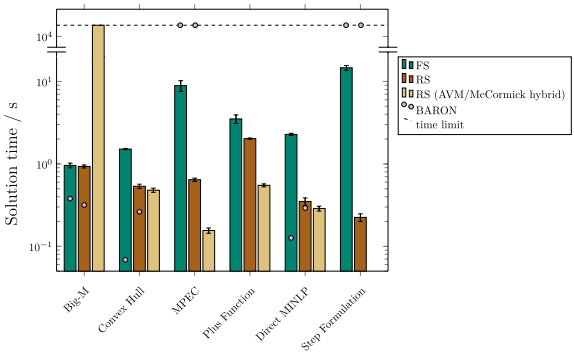

Fig. 7 Solution time for Problem (HEXGDP) using the problem formulations presented in Sect. 3 and the Step Formulation. As BARON can not handle *max*-functions, there are no results for the Plus Function formulation. The solution times of the optimization with BARON and the MPEC and Step Formulation exceed the maximum solution time of 86400 s. No results are given for the RS (AVM/McCormick hybrid) Step Formulation, as it does not contain repeated nonlinear terms that could be replaced by AVs. The error bars represent the standard deviation from the arithmetic mean value of the solution time from 100 optimization runs using MAiNGO

6 Conclusion

Superstructure optimization problems for process synthesis often contain nonconvex functions resulting in nonconvex MINLP problems, for which global solvers need to be used. Although the same problem formulations as those developed for superstructure optimization problems with only convex functions can be used, it remains unclear whether these are always the most computationally efficient ones for problems with nonconvex functions. In particular, for problems with only convex functions, preference is usually given to formulations such as Big-M or Convex Hull that avoid introducing nonconvexities, thus resulting in a convex MINLP or even MILP. We conjectured that for problems that contain nonconvex functions anyway, alternative formulations that do introduce additional nonconvexities but may result in smaller problems or allow tighter relaxations could be promising.

Our analysis shows that for problems containing nonconvex functions anyway, these additional nonconvexities do not necessarily have a considerable negative influence on the optimization. The resulting relaxations often seem to remain comparably tight despite these additional nonconvexities and the corresponding problem formulations contain fewer variables than the conventional formulations. Accordingly, the alternative formulations result in similar or even lower computational time than the conventional ones for most considered examples. As an additional approach to reduce problem size, we exploited reduced-space (RS) formulations, where we eliminate as many optimization variables as possible using the model equations. Despite weaker relaxations, the smaller problem size of RS formulations is beneficial for all problem formulations for the considered case studies.

The comparison of the results obtained with our open-source solver MAiNGO with those obtained with the state-of-the-art commercial solver BARON shows that neither the auxiliary variable method (AVM) employed in BARON, nor the propagation of McCormick relaxations employed in MAiNGO suffers significantly from the additional nonconvex terms resulting from the unconventional problem reformulation approaches. Yet, the reduction of the problem size in the RS formulations can be exploited more effectively by MAiNGO, resulting in the lowest overall solution times observed for the illustrative example problems. For the conventional reformulation approaches, BARON generally outperforms MAiNGO.

We have extended the comparison of model formulations by also considering a problem with piecewise-defined functions instead of unit selections, which can be formulated as a superstructure optimization problem. For this problem, the RS formulation for all reformulation approaches was again computationally more efficient than the FS formulation. A formulation that directly treated the discontinuous function with the help of step functions was particularly advantageous.

In summary, the unconventional reformulation approaches, which introduce nonconvex terms, are promising for nonconvex superstructure optimization problems, especially when combined with RS formulations. Bearing in mind that

modeling using these approaches is rather intuitive and the resulting models simple, they represent an interesting alternative to established approaches such as the Big-M and Convex Hull method. To further validate this claim, a benchmark library with more complex superstructure optimization problems is required for such comparative analyses in future studies. In this work, only a few simple flowsheet optimization problems have been considered, as each problem formulation needed to be implemented manually for all reformulation approaches and for both the FS and RS formulation individually. To overcome the considerable manual effort and benefit from a flexible application of different reformulation approaches given specific problem characteristics, the automated generation of problem formulations (in the vein of, e.g., the open-source Python package for component-oriented modeling and optimization for nonlinear design and operation of integrated energy systems COMANDO (Langiu et al. 2021), the Pyomo.GDP package (Chen et al. 2018) that allows automated reformulation and solution of GDPs, or the Pyosyn framework (Chen et al. 2021) for superstructure modeling) based on a GDP problem would be beneficial.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11081-021-09707-y.

Acknowledgements This work has received funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) “Improved McCormick Relaxations for the efficient Global Optimization in the Space of Degrees of Freedom" [MI 1851/4-1]. The authors also gratefully acknowledge funding by the German Federal Ministry of Education and Research (BMBF) within the project NAMOSYN: Nachhaltige Mobilität durch synthetische Kraftstoffe [03SF0566P0].

Author Contributions J. Burre developed the concept of the study, performed the optimization, and wrote the majority of the manuscript. J. Burre and D. Bongartz analyzed the results. A. Mitsos proposed the key idea of the study and supervised the work. All authors contributed to the discussions and revisions of the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. This work has received funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) “Improved McCormick Relaxations for the efficient Global Optimization in the Space of Degrees of Freedom" [MI 1851/4-1] and from the German Federal Ministry of Education and Research (BMBF) within the project NAMOSYN: Nachhaltige Mobilität durch synthetische Kraftstoffe [03SF0566P0].

Availability of data and material Not applicable.

Declarations

Conflicts of interest There are not conflicts to declare.

Code availability All problem formulations are available open-source within the attached *zip*-archive.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Achterberg T (2009) SCIP: solving constraint integer programs. Math Program Comput 1(1):1–41. https://doi.org/10.1007/s12532-008-0001-1

Ahmetović E, Grossmann IE (2011) Global superstructure optimization for the design of integrated process water networks. AIChE J 57(2):434–457. https://doi.org/10.1002/aic.12276

Baumrucker B, Renfro J, Biegler L (2008) MPEC problem formulations and solution strategies with chemical engineering applications. Comput Chem Eng 32(12):2903–2913. https://doi.org/10.1002/aic.12276

Ben-Tal A, Eiger G, Gershovitz V (1994) Global minimization by reducing the duality gap. Math Program 63(1–3):193–212. https://doi.org/10.1007/bf01582066

Bongartz D (2020) Deterministic global flowsheet optimization for the design of energy conversion processes. PhD thesis, RWTH Aachen University

Bongartz D, Mitsos A (2017) Deterministic global optimization of process flowsheets in a reduced space using McCormick relaxations. J Global Optim 69(4):761–796. https://doi.org/10.1007/s10898-017-0547-4

Bongartz D, Mitsos A (2019) Deterministic global flowsheet optimization: Between equation-oriented and sequential-modular methods. AIChE J 65(3):1022–1034. https://doi.org/10.1007/s10898-017-0547-4

Bongartz D, Najman J, Sass S, Mitsos A (2018) MAiNGO – McCormick-based Algorithm for mixed-integer Nonlinear Global Optimization. Tech. rep., Process Systems Engineering (AVT.SVT), RWTH Aachen University, http://permalink.avt.rwth-aachen.de/?id=729717

Burre J, Bongartz D, Brée L, Roh K, Mitsos A (2020) Power-to-x: Between electricity storage, e-production, and demand side management. Chem Ing Tec 92(1–2):74–84. https://doi.org/10.1002/cite.201900102

Burre J, Bongartz D, Deutz S, Mebrahtu C, Osterthun O, Sun R, Völker S, Bardow A, Klankermayer J, Palkovits R, Mitsos A (2021) Comparing pathways for electricity-based production of dimethoxymethane as a sustainable fuel. Energy Environ Sci 14(7):3686–3699. https://doi.org/10.1039/d1ee00689d

Byrne RP, Bogle IDL (2000) Global optimization of modular process flowsheets. Ind Eng Chem Res 39(11):4296–4301. https://doi.org/10.1021/ie990619d

Ceria S, Soares J (1999) Convex programming for disjunctive convex optimization. Math Program 86(3):595–614. https://doi.org/10.1007/s101070050106

Chachuat B, Houska B, Paulen R, Peric N, Rajyaguru J, Villanueva ME (2015) Set-theoretic approaches in analysis, estimation and control of nonlinear systems. IFAC-PapersOnLine 48(8):981–995. https://doi.org/10.1016/j.ifacol.2015.09.097

Chen C, Mangasarian OL (1996) A class of smoothing functions for nonlinear and mixed complementarity problems. Comput Optim Appl 5(2):97–138. https://doi.org/10.1007/bf00249052

Chen Q, Johnson ES, Siirola JD, Grossmann IE (2018) Pyomo.GDP: Disjunctive models in python. In: 13th International symposium on process systems engineering (PSE 2018), Elsevier, pp 889–894, https://doi.org/10.1016%2Fb978-0-444-64241-7.50143-9

Chen Q, Liu Y, Seastream G, Siirola JD, Grossmann IE (2021) Pyosyn: a new framework for conceptual design modeling and optimization. Comput Chem Eng 153:107414. https://doi.org/10.1016/j.compchemeng.2021.107414

Dür M, Horst R (1997) Lagrange duality and partitioning techniques in nonconvex global optimization. J Optim Theory Appl 95(2):347–369. https://doi.org/10.1023/A:1022687222060
Epperly TGW, Pistikopoulos EN (1997) A reduced space branch and bound algorithm for global optimization. J Global Optim 11(3):287–311. https://doi.org/10.1023/A:1008212418949
Furman KC, Sawaya NW, Grossmann IE (2020) A computationally useful algebraic representation of nonlinear disjunctive convex sets using the perspective function. Computational Optimization and Applications pp 1–26
Gopal V, Biegler LT (1999) Smoothing methods for complementarity problems in process engineering. AIChE J 45(7):1535–1547. https://doi.org/10.1002/aic.690450715
Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3(3):227–252. https://doi.org/10.1023/A:1021039126272
Grossmann IE, Lee S (2003) Generalized convex disjunctive programming: Nonlinear convex hull relaxation. Comput Optim Appl 26(1):83–100. https://doi.org/10.1023/A:1025154322278
Grossmann IE, Trespalacios F (2013) Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming. AIChE J 59(9):3276–3295. https://doi.org/10.1002/aic.14088
Huster WR, Schweidtmann AM, Lüthje JT, Mitsos A (2020) Deterministic global superstructure-based optimization of an organic rankine cycle. Comput Chem Eng. https://doi.org/10.1016/j.compchemeng.2020.106996
Kannan R, Barton PI (2017) Convergence-order analysis of branch-and-bound algorithms for constrained problems. J Global Optim 71(4):753–813. https://doi.org/10.1007/s10898-017-0532-y
Kılınç MR, Sahinidis NV (2017) Exploiting integrality in the global optimization of mixed-integer nonlinear programming problems with BARON. Opt Methods Softw 33(3):540–562. https://doi.org/10.1080/10556788.2017.1350178
Langiu M, Shu DY, Baader FJ, Hering D, Bau U, Xhonneux A, Müller D, Bardow A, Mitsos A, Dahmen M (2021) COMANDO: a next-generation open-source framework for energy systems optimization. Comput Chem Eng. https://doi.org/10.1016/j.compchemeng.2021.107366
Lee S, Grossmann IE (2000) New algorithms for nonlinear generalized disjunctive programming. Comput Chem Eng 24(9–10):2125–2141. https://doi.org/10.1016/S0098-1354(00)00581-0
Lee S, Grossmann IE (2003) Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks. Comput Chem Eng 27(11):1557–1575. https://doi.org/10.1016/S0098-1354(03)00098-X
Luo ZQ, Pang JS, Ralph D (1996) Mathematical programs with equilibrium constraints. Cambridge University Press. https://doi.org/10.1017/cbo9780511983658
McCormick GP (1976) Computability of global solutions to factorable nonconvex programs: Part i – convex underestimating problems. Math Program 10(1):147–175. https://doi.org/10.1007/bf01580665
Mencarelli L, Chen Q, Pagot A, Grossmann IE (2020) A review on superstructure optimization approaches in process system engineering. Comput Chem Eng. https://doi.org/10.1016/j.compchemeng.2020.106808
Misener R, Floudas CA (2014) ANTIGONE: Algorithms for coNTinuous / integer global optimization of nonlinear equations. J Global Optim 59(2–3):503–526. https://doi.org/10.1007/s10898-014-0166-2
Mitsos A, Chachuat B, Barton PI (2009) McCormick-based relaxations of algorithms. SIAM J Optim 20(2):573–601. https://doi.org/10.1137/080717341
Najman J, Bongartz D, Mitsos A (2021) Linearization of McCormick relaxations and hybridization with the auxiliary variable method. J Global Optim 80(4):731–756. https://doi.org/10.1007/s10898-020-00977-x
Nemhauser G, Wolsey L (1988) Integer and combinatorial optimization. Wiley, USA
Raman R, Grossmann I (1994) Modelling and computational techniques for logic based integer programming. Comput Chem Eng 18(7):563–578. https://doi.org/10.1016/0098-1354(93)E0010-7
Roh K, Bardow A, Bongartz D, Burre J, Chung W, Deutz S, Han D, Heßelmann M, Kohlhaas Y, König A, Lee JS, Meys R, Völker S, Wessling M, Lee JH, Mitsos A (2020) Early-stage evaluation of emerging CO2 utilization technologies at low technology readiness levels. Green Chem 22(12):3842–3859. https://doi.org/10.1039/c9gc04440j
Ruiz JP, Grossmann IE (2010) Strengthening of lower bounds in the global optimization of bilinear and concave generalized disjunctive programs. Comput Chem Eng 34(6):914–930. https://doi.org/10.1016/j.compchemeng.2009.10.016
Sahlodin AM, Watson HAJ, Barton PI (2016) Nonsmooth model for dynamic simulation of phase changes. AIChE J 62(9):3334–3351. https://doi.org/10.1002/aic.15378

Sawaya N (2006) Reformulations, relaxations and cutting planes for generalized disjunctive programming. PhD thesis, Carnegie Mellon University

Scholtes S (2001) Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. SIAM J Optim 11(4):918–936. https://doi.org/10.1137/s1052623499361233

Schweidtmann AM, Mitsos A (2018) Deterministic global optimization with artificial neural networks embedded. J Optim Theory Appl 180(3):925–948. https://doi.org/10.1007/s10957-018-1396-0

Schweidtmann AM, Bongartz D, Grothe D, Kerkenhoff T, Lin X, Najman J, Mitsos A (2021) Deterministic global optimization with gaussian processes embedded. Math Program Comput. https://doi.org/10.1007/s12532-021-00204-y

Scott JK, Stuber MD, Barton PI (2011) Generalized McCormick relaxations. J Global Optim 51(4):569–606. https://doi.org/10.1007/s10898-011-9664-7

Singh A, Forbes J, Vermeer P, Woo S (2000) Model-based real-time optimization of automotive gasoline blending operations. J Process Control 10(1):43–58. https://doi.org/10.1016/S0959-1524(99)00037-2

Smith EM, Pantelides CC (1997) Global optimisation of nonconvex MINLPs. Comput Chem Eng 21:S791–S796. https://doi.org/10.1016/S0098-1354(97)87599-0

Stuber MD, Scott JK, Barton PI (2014) Convex and concave relaxations of implicit functions. Opt Methods Softw 30(3):424–460. https://doi.org/10.1080/10556788.2014.924514

Tawarmalani M, Sahinidis NV (2002) Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming. Springer, USA

Trespalacios F, Grossmann IE (2015) Improved big-m reformulation for generalized disjunctive programs. Comput Chem Eng 76:98–103. https://doi.org/10.1016/j.compchemeng.2015.02.013

Tsoukalas A, Mitsos A (2014) Multivariate McCormick relaxations. J Global Optim 59(2–3):633–662. https://doi.org/10.1007/s10898-014-0176-0

Türkay M, Grossmann IE (1996) Disjunctive programming techniques for the optimization of process systems with discontinuous investment costs-multiple size regions. Ind Eng Chem Res 35(8):2611–2623. https://doi.org/10.1021/ie9600856

Türkay M, Grossmann IE (1996) Logic-based MINLP algorithms for the optimal synthesis of process networks. Comput Chem Eng 20(8):959–978

Wechsung A, Barton PI (2013) Global optimization of bounded factorable functions with discontinuities. J Global Optim 58(1):1–30. https://doi.org/10.1007/s10898-013-0060-3

Wechsung A, Scott JK, Watson HAJ, Barton PI (2015) Reverse propagation of McCormick relaxations. J Global Optim 63(1):1–36. https://doi.org/10.1007/s10898-015-0303-6