SAND2022-15154C

Optimization-based Approaches for Design of Chemical Process Families Using ReLU Surrogates

Georgia Stinchfield[a], Lorenz T. Biegler[a], John C. Eslick[d,e], Clas Jacobson[c], David C. Miller[d], John D. Siirola[f], Miguel A. Zamarripa[d,e], Chen Zhang[c], Qi Zhang[b], Carl D. Laird[*a]

[a] Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213
[b] Dept. of Chem. Eng. and Materials Science, U. of Minnesota, Minneapolis, MN 55455
[c] Carrier Global Corporation, Palm Beach Gardens, FL 33418
[d] National Energy Technology Laboratory, Pittsburgh, PA, USA
[e] NETL Support Contractor, Pittsburgh, PA, USA
[f] Sandia National Laboratories, P.O. Box 5800, Albuquerque, NM 87123, United States

Abstract

Tackling national climate change and infrastructure goals requires the widespread deployment of process technologies across a large number of decentralized sites with different geographical, environmental, and operational requirements. A conventional engineering design approach, which would develop unique designs for each potential installation, is time-consuming and misses valuable opportunities for shared design effort across all the potential installations. To provide accelerated deployment and simultaneously reduce engineering and manufacturing costs across all installations, we propose a computationally efficient approach for the simultaneous design of a family of processes. With this approach, we define the process family as a set of potential installations (or products) where each of these installations has specific design requirements that need to be met. We then formulate and solve an optimization problem to simultaneously design each of the processes in the family while exploiting the opportunity for shared designs of selected units or components across the process family. This approach reduces the time to market, lowers the manufacturing cost of components by exploiting economies of numbers and learning, and decreases the up-front engineering design effort. Our initial formulation is a nonlinear generalized disjunctive program that seeks to determine the sizes of each of the common units to manufacture, while simultaneously solving for the assignment of units to each installation in a way that is feasible and cost optimal. Given the computational challenges associated with this formulation, we explore two alternative mixed-integer linear programming (MILP) formulations, one that is based on full-discretization of the design space and another that uses ReLU activated neural network models as surrogates for the cost and performance functions. ReLU neural networks can be represented exactly within mixed-integer linear programming formulations, and we use OMLT to import these surrogates into the optimization framework Pyomo. We demonstrate the computational benefits of these formulations on a family of transcritical CO_2 refrigeration cycles to meet a range of design criteria over both cooling capacity and outside air temperature. Our surrogate-based approach produces designs similar to the full-discretization approach with an order of magnitude reduction in the number of required simulations to produce the input data.

Keywords

Process Family Design, Manufacturing, Optimization, Machine Learning, Neural Networks, Surrogates.

Introduction

The transformation of the chemical process industry to tackle environmental and climate concerns requires rapid, widespread deployment of new process technologies across a wide range of industrial sites. While research has focused on development of new technologies and flowsheet optimization, economic deployment of these technologies remains a challenge. For example, Wilberforce et al. (2019) indicate that the biggest challenge surrounding mass deployment of post-combustion carbon capture systems is the cost of development and installation for a particular carbon-emitting site; the technology itself is mature and has been refined for decades (Wilberforce et al., 2019). Traditional process design approaches, focused on achieving economies of scale, seek to independently optimize the design of each installation, resulting in significant engineering effort and manufac-

[1] Corresponding author. Email: clairdandrew.cmu.edu

turing costs associated with a large number of unique, individual designs. This traditional approach is difficult to scale for widespread deployment across many decentralized installations.

Profit-enhancing and timeline-reducing opportunities are present in both engineering design and manufacturing. Engineering design costs can be estimated to be approximately 10% (larger projects) and 30% (smaller projects) of the cost of the plant plus the cost of modifications and improvements (Towler and Sinnott, 2022). Furthermore, manufacturing a large number of units that are unique for each installation is inefficient and misses the opportunity to exploit economies of learning and reduce per-unit manufacturing costs. Modularity has been well-studied in the process systems engineering community and can be used to reduce engineering and manufacturing costs. However, the approach of *numbering-up* fails to fully exploit economies of scale and can lead to sub-optimal designs.

We consider instead the design of multiple processes simultaneously, rather than as separate design tasks, building on ideas from the areas of product family design and platform development. Companies such as Nissan and Renault (Sanchez and Shibata, 2021), and Boeing, Ford, and Caterpillar (Simpson et al., 2014) have expanded their product variety to draw in customers while reducing manufacturing and design costs, relying on common sub-components or units shared throughout a set of similar products. We extend these ideas and present new optimization-based approaches for the simultaneous design of chemical process families that can exploit opportunities for shared components while covering a range of design requirements for a large number of potential installations. The paper is outlined as follows. We first present background on modularity and product family design, and we discuss the concept of optimal process family design. We then describe three optimization formulations for process family design. We compare the computational performance and solution quality of two of these formulations using a case study based on a model for a transcritical CO_2 refrigeration cycle from the literature (Li and Groll, 2005). The model is formulated within the IDAES platform (Lee et al., 2021) and used to produce the required input data for the process family design formulations.

Background

Modular manufacturing mass-produces a select set of relatively small units that can be used independently or "numbered up" to meet capacity goals (meaning more than one modular unit is used in parallel to achieve a desired capacity or operating conditions). Baldea et al. (2017) shows that modularity significantly decreases construction time and cost of a chemical process. In one case study, the total cost of a project was reduced by approximately 30% while another case study reported that the 3-year project timeline was reduced by roughly 1 year. Benefits of modularization include the reduction of manufacturing costs through economies of learning and a decrease in detailed engineering design effort (since the number of unique unit designs is reduced).

While modularity has significant benefits, it can reduce design flexilibity and may not sufficiently exploit economies of scale. Chen and Grossmann (2019) mathematically study trade-off factors between a modular process and a conventional plant. In their case studies, the authors found that modularity could decrease costs and improve profits compared to a conventional design approach but not in every design scenario and, notably, not by an exceptionally large profit margin. Arora et al. (2020) proposed a methodology to determine which units in multiple, similar chemical processes should be designed modularly versus uniquely based on similar tradeoffs. Bhosekar and Ierapetritou (2020) evaluated optimization of modular design frameworks under variablity in demands, using flexibility analysis based on machine learning. Much of the work in modularization of process systems focuses on analyzing trade-offs between up-front cost savings and long-term economic impact.

Product family design, on the other hand, seeks to balance cost savings from manufacturing standardization with satisfaction of variety in customer demands (Simpson et al., 2014). Product family design is different from modularity; it produces a *platform* of components with designs that are distributed across a *family* of similar products (i.e., without the need for stacking or numbering up). The platform of components is customized to match customer demand for variety in a particular product. Some products require more variety to satisfy wide market demands while others stress more commonality to exploit additional manufacturing savings. Finding the optimal tradeoff between degree of commonality and level of variety for a particular product is a central area of product family design research.

Product family design has seen significant success in industrial applications within, for example, the automotive and aviation industries. Thonemann and Brandeau (2000) investigated methods for achieving optimal commonality in manufacturing products applied to a wire-harness design problem faced by a major automobile manufacturer. Sanchez and Shibata (2021) describe the partnership between major automotive companies Nissan and Renault. Both companies achieved significant cost reductions while maintaining distinctive brand identities and product variety by following a joint product family design manufacturing scheme. Baud-Lavigne et al. (2016) investigated simultaneously designing a product family and the resulting supply chain, considering trade-offs between construction of the bill of materials and design of the supply chain network.

Optimization Formulations for Process Family Design

Product family design brings important ideas for the design of chemical processes intended for widespread manufacturing and deployment (e.g., decentralized technologies to address climate change goals). Simpson et al. (2014) define a product family to be a "set of products that share one or more common 'elements'... yet target a variety of different market segments." Similarly, we define a process family as a set of processes that may share one or more common unit designs while collectively targeting a range of boundary conditions and performance requirements. Effective process family design allows us to exploit economies of learning similar to modularity, but it also offers increased design flexibility and the ability to further exploit economies of scale.

In this section, we propose computationally tractable optimization formulations for the simultaneous design of a family of processes to collectively meet a range of potential performance requirements. We call each process in the family an *installation* since each could represent a particular installation site in a planned multi-site deployment (e.g., across a large company). Note that an *installation* could also represent a particular product specification, where multiple instances of each product are to be manufactured and sold by a process vendor. Each installation $i \in I$ is defined (and differentiated) by the specific core design requirements (e.g., desired capacity, expected boundary conditions, etc.). To reduce design and manufacturing costs, we exploit opportunities for shared design of sub-components, referred to as *units*, across the process family. We identify the set $k \in K$ that contains the units that are candidates for shared design (e.g., primary exchanger, packed bed reactor). Strong candidates are those that drive the total purchase and operating costs of the system but do not have a widely available set of existing market options (i.e., they would typically be built-to-order). We develop a framework to simultaneously optimize the collective set of installations $i \in I$ in the process family while also solving for the best unit designs for each unit $k \in K$. The key decisions are (1) the number of common designs to consider for each unit, (2) the specific design variables for each of these unit designs, (3) the allocation of these common unit designs to each of the installations in the family, and (4) the design and operating variables that are unique to each installation. These decisions are selected so that the entire process family is cost optimal.

We first formulate this problem as a nonlinear generalized disjunctive programming problem. Given the computational challenges associated with the solution of large-scale mixed-integer nonlinear programming (MINLP) problems, we explore two alternative solution strategies. The cost and performance of a particular installation is dependent on the design variable values for the shared units and the design and operating variables that can be uniquely optimized for that installation. To develop our first MILP formulation, we create a discrete set of candidate sizes for each of the common units. Then, for each installation, we form the set of all potential alternatives based on the cartesian product of the candidate sizes for each common unit. For each of these alternatives, we optimize over the remaining variables to minimize the annualized cost of each of the alternatives. These solutions form the input data for the combinatorial MILP formulation that seeks to find the best designs for common units and the optimal allocation of those common units across the installations. The bottleneck for this approach is the optimization of the large number of discretized combinations for each installation. To further reduce the computational effort, we also develop an MILP formulation that uses neural network models with rectifying linear unit (ReLU) activation functions as surrogates for the cost and performance of the system. ReLU neural networks are selected since they are exactly representable within MILP formulations (Grimstad and Andersson, 2019; Fischetti and Jo, 2018; Anderson et al., 2020), allowing us to remove the nonlinearities and write the entire formulation as an MILP. ReLU networks can be easily translated into Pyomo (Bynum et al., 2021) models using the newly released Python package, OMLT (Ceccon et al., 2022). The goal of this approach is to find similar (or improved) solutions with surrogates that require significantly less input data than the full-discretization formulation. We now describe these three formulations in detail. Parameters, variables, and set definitions are shown in Table 1.

Table 1: Parameters, Variables, and Sets

	Description
$i \in I$	Set of installation sites or processes in the family
$k \in K$	Set of candidate units with common designs
w_i	Weighting parameter for installation i
c_i	Annualized CAP/OPEX of install. i
$c_{i,a}$	Annualized CAP/OPEX for alt. a with install. i
b_i	Design specifications for installation i
u_i	Operating variables for installation i
$d_{i,k}$	Design variables for unit k in installation i
v_i	Installation specific design variables
$j \in J_k$	Set of indices of unit design j for each $k \in K$
$\hat{d}_{k,j}$	Design variables for unit design j of unit k
Y_{ikj}	True if unit design j selected for unit k in i
$x_{i,a}$	Variable: one if alternative a selected for i
$s \in S_k$	Set of discretized design candidates s for unit k
$z_{k,s}$	True if unit design candidate s from unit k is selected for manufacture
N_k	Number of designs to be manufactured for unit k
$a \in A_i$	Set of alternatives available for each installation i
p_i	Performance indicators for installation i

Formulation 1: Nonlinear GDP Formulation

Here, we present a nonlinear generalized disjunctive programming formulation for the process family design problem. Note, we do not solve this directly, but use it as a basis for the subsequent formulations.

$$\min_{u,d,\hat{d},v,y,c} \sum_{i\in I} w_i c_i \tag{1a}$$

$$\text{s.t. } h(b_i,u_i,d_{i,1},...,d_{i,k},v_i) = 0 \qquad \forall i \in I \tag{1b}$$

$$p_i^L \le p(b_i,u_i,d_{i,1},...,d_{i,k},v_i) \le p_i^U \qquad \forall i \in I \tag{1c}$$

$$\bigvee_{j\in J_k} \begin{bmatrix} Y_{ikj} \\ d_{i,k} = \hat{d}_{k,j} \end{bmatrix} \qquad \forall i \in I, k \in K \tag{1d}$$

$$c_i = f_i^c(b_i,u_i,d_{i,1},...,d_{i,k},v_i) \qquad \forall i \in I \tag{1e}$$

$$d_{i,k}^L \le d_{i,k} \le d_{i,k}^U \qquad \forall i \in I, k \in K \tag{1f}$$

$$v_i^L \le v_i \le v_i^U \qquad \forall i \in I, k \in K \tag{1g}$$

$$Y_{ikj} \in \{\text{True}, \text{False}\} \qquad \forall i \in I, k \in K, j \in J_k \tag{1h}$$

The objective is to minimize the weighted sum of annualized capital and operating costs for the process family, as shown in Eq. (1a). Eq. (1b) represents the process model, and Eq. (1c) represents any performance constraints for the installations. The parameters b_i are design specifications or boundary conditions specific to installation i; u_i are operating variables for installation i (allowed to vary for each installation to meet performance targets); $d_{i,k}$ are design variables

(e.g., sizes) for shared units k in installation i; and ν_i are design variables whose values are allowed to be unique for each installation i. In some cases, the design requirements for an installation i may be fixed (through b_i) while others may be represented as inequalities given by the performance constraints Eq. (1c). For each unit, we allow a certain number of unique unit designs that can be used across the family, captured in the sets $j \in J_k$ for each unit k. The $\hat{d}_{k,j}$ variables represent the values of the design variables for the optimized design j of unit k. The disjunctions in Eq. (1d) determine which of the common unit designs are selected for each installation i, where Y_{ikj} is True if installation i selects unit design j for unit k. If selected, the equations in the disjunct ensure that the design variables $d_{i,k}$ are made equivalent to $\hat{d}_{k,j}$. Here, we use $\veebar$ to indicate that only one of the disjuncts in each disjunction is selected. Eq. (1e) calculates annualized capital and operating cost based on which unit designs are selected. Although this formulation can be converted to an MINLP (e.g., Big-M or convex-hull transformation), these large-scale MINLPs would be difficult to solve for practical problems. Therefore, we explore two alternative formulations provided below.

Formulation 2: Discretized Formulation

Here, we formulate an MILP for the process family design problem by discretizing the design space for the common units. In particular, we decide on a set of candidate sizes, $s \in S_k$ for each of the common units k. Then, for each installation i, we perform an optimization over ν_i and u_i for each combination of candidate unit sizes to determine optimal costs, feasibility, and performance . We call each of these combinations a design *alternative*, and we define the set of all feasible alternatives for each installation i as $a \in A_i$. An alternative is feasible for installation i if it meets all the the desired performance requirements. We show the discretized formulation of the problem, as first presented by Zhang et al. (2022).

$$\min_{x,z} \sum_{i\in I} w_i \sum_{a\in A_i} c_{i,a}x_{i,a} \tag{2a}$$

$$\text{s.t.} \sum_{s\in S_k} z_{k,s} \le N_k \qquad \forall k \in K \tag{2b}$$

$$\sum_{a\in A_i} x_{i,a} = 1 \qquad \forall i \in I \tag{2c}$$

$$x_{i,a} \le z_{k,s} \qquad \forall i \in I, a \in A_i, (k,s) \in Q_a \tag{2d}$$

$$0 \le x_{i,a} \le 1 \qquad \forall i \in I, a \in A_i \tag{2e}$$

$$z_{k,s} \in \{0,1\} \qquad \forall k \in K, s \in S_k \tag{2f}$$

We have a set of candidate sizes $s \in S_k$ for each unit type k, and the binary variable $z_{k,s}$ is one if the optimizer chooses to manufacture size s, and zero otherwise. The variable $x_{i,a}$ indicates if we choose alternative a for installation i. The objective is to minimize the weighted sum of annualized costs as shown in Eq. (2a). Eq. (2b) ensures the total number of selected unit designs for k does not exceed N_k (the total number of size options we wish to offer in the process family). Eq. (2c) ensures exactly one alternative is selected for each installation i. Eq. (2d) ensures that alternative a can only be selected if we have chosen to manufacture the unit sizes needed by that alternative. Eq. (2e) represents the bounds on the decision variable $x_{i,a}$. Though the decision variables $x_{i,a}$ are bounded between 0 and 1, under the mild assumption that each cost, $c_{i,a}$, is different from one another, this formulation converges to values of 0 or 1 at optimality.

Formulation 3: ReLU Surrogates Formulation

While optimization of the discretized formulation above is fast, significant pre-computation time is required to perform the optimizations for each potential alternative. In the new formulation described here, we consider the original GDP (Formulation 1) and replace the nonlinear models in Eqs. (1b,1c) with piecewise linear surrogates. We can construct these surrogates using far less data than that required for the full-discretization formulation, resulting in a significant computational improvement.

We use neural network (NN) models with ReLU activation functions for the surrogates. These are multivariate piecewise linear models that can be represented exactly within MILP formulations (Grimstad and Andersson, 2019). The Python package OMLT (Ceccon et al., 2022) allows us to import these NN surrogates into Pyomo models. Furthermore, there are strong training tools for ReLU neural networks (e.g., Keras) that simplify surrogate construction and interface directly to OMLT.

$$\min_{d,\hat{d},y} \sum_{i\in I} w_i c_i \tag{3a}$$

$$\text{s.t. } f_i^c(c_i, b_i, d_{i,1}, ..., d_{i,k}) = 0 \qquad \forall i \in I \tag{3b}$$

$$f_i^p(p_i, b_i, d_{i,1}, ..., d_{i,k}) = 0 \qquad \forall i \in I \tag{3c}$$

$$\bigvee_{j\in J_k} \begin{bmatrix} Y_{ikj} \\ d_{i,k} = \hat{d}_{k,j} \end{bmatrix} \qquad \forall i \in I, k \in K \tag{3d}$$

$$\hat{d}^L_{k,j} \le \hat{d}_{k,j} \le \hat{d}^U_{k,j} \qquad \forall k \in K, j \in J_k \tag{3e}$$

$$p_i^L \le p_i \le p_i^U \qquad \forall i \in I \tag{3f}$$

$$c_i \ge 0 \qquad \forall i \in I, k \in K \tag{3g}$$

$$Y_{ikj} \in \{\text{True}, \text{False}\} \qquad \forall i \in I, k \in K, j \in J_k \tag{3h}$$

The objective, Eq. (3a), is again the weighted sum of annualized costs. Eq. (3b) represents the MILP constraints from OMLT that define the surrogate for the annualized cost. For this trained NN, the boundary conditions and unit design variables are the inputs, and the annualized cost is the output. Likewise, Eq. (3c) represents the MILP constraints of the surrogate for the performance indicators. The disjunctions, shown in Eq. (3d), are the same as those in Eq. (1d). Eq. (3f) ensures that we do not select unit designs for installation i that do not satisfy the bounds on the performance indicator.

Numerical Case Study

To demonstrate the computational performance and effectiveness of these formulations, we consider the design of a family of products for a transcritical CO_2 refrigeration cycle, described by Li and Groll (2005) and modeled using IDAES (Lee et al., 2021). The design requirements that were varied for each installation were the required cooling capacity and outside air temperature. Desired cooling capacity is based on needs at individual installations. The maximum outside air

temperature used for design is directly affected by geographical location. We considered seven capacities and eight outside air temperatures, resulting in 56 unique installation requirements. Three units were considered for shared design in the process family, including the compressor, evaporator, and condenser. For the fully discretized approach (Formulation (2)), we considered 10 evaporator sizes, 14 condenser sizes, and 10 compressor sizes. For this example, we do not consider additional installation specific optimization variables u_i and v_i. Therefore, we can perform simulations for each alternative to determine costs and performance.This required 78,400 simulations and approximately 1120 min. of computational time. For Formulation (3), we sampled 7,840 points from the discretized space to represent an order of magnitude fewer simulations. For the NN architectures, we selected a single ReLU layer with 15 nodes . The cost NN was trained on 4,132 points (all feasible alternatives) in 199 epochs. The performance NN trained on all 7,840 points in 2,361 epochs.

Both the discretized formulation and the surrogates formulation of the process family design problem were programmed in Pyomo (Bynum et al., 2021) and solved using Gurobi (Optimization, LLC Gurobi et al., 2020). The piecewise linear surrogate formulation required two additional Python-based software packages: Optimization and Machine Learning Toolkit (OMLT) (Ceccon et al., 2022), which represents machine learning models within the Pyomo optimization environment, and a Pyomo extension, Pyomo.GDP (Chen et al., 2022), to represent the disjunctions.

We restrict the optimization to select only two sizes of each of the three units (evaporator, condenser, and compressor) for manufacturing. Figures 1 and 2 show the optimal allocation of unit designs for each installation. Table 2 lists the the optimal unit designs and Table 3 lists computational times and problem sizes.

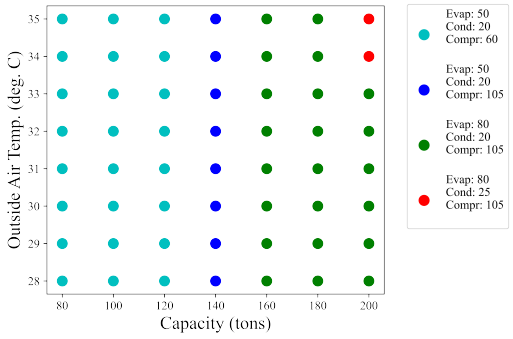

Figure 1: Discretized (Formulation 2) Solution

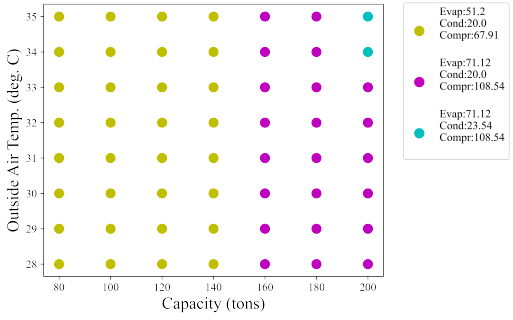

Figure 2: ReLU Surrogates (Formulation 3) Solution

Table 2: Design Decisions of Optimization

Variable	Discrete Sol.	Surrogate Sol.
Evaporator, Design 1	50 m^2	51.2 m^2
Evaporator, Design 2	80 m^2	71.12 m^2
Condenser, Design 1	20 m^2	20 m^2
Condenser, Design 2	25 m^2	23.54 m^2
Compressor, Design 1	60 $\frac{mol}{s}$	67.91 $\frac{mol}{s}$
Compressor, Design 2	105 $\frac{mol}{s}$	108.54 $\frac{mol}{s}$
Total Annualized Cost	$4,173,030	$4,056,984

Table 3: Solution Comparison

Attribute	Discrete Sol.	Surr. Sol.
Simulation Time	67,200 sec.	6,720 sec.
Training Time	-	600 sec.
Gurobi Solve Time	14.66 sec.	71.70 sec.
Total Comp. Time	67,214.66 sec.	7391.7 sec.
Num. of Constraints	116,184	11,540
Num. of Cont. Vars.	38,705	5775
Num. of Binary Vars.	34	2016

The surrogates formulation approximates the continuous design space and allows for more specific size selections. For example, since the surrogates formulation is allowed to select a compressor of size 67.91 $\frac{\text{mol}}{\text{s}}$, this slightly larger compressor is able to address all of the installations at a capacity of 140 tons, whereas the discretized formulation has a fixed granularity and selected a size of 60 $\frac{\text{mol}}{\text{s}}$ for the lower ranges but required a larger compressor (105 $\frac{\text{mol}}{\text{s}}$) to ensure feasibility at 140 tons. Another trade-off is observed in the evaporator designs. The discretized formulation allocated an evaporator of size 80 m^2 to capacities 160, 180, and 200 tons while the surrogates formulation selected size 71.12 m^2 for the same region, allowing for a smaller unit design. Most importantly, the surrogates formulation required an order of magnitude less input data with a corresponding reduction in computational time and problem size, as shown in Table 3. Also note that we verified the surrogates solution with the rigorous model, and all designs were feasible while the optimal cost had approximately 2% error.

Conclusions and Future Work

In this work, we developed multiple formulations for the process family design problem and applied two of them to a case study of a transcritical CO_2 refrigeration system. One formulation used full discretization of the design space, and one was based on the use of neural network surrogates to replace nonlinear model equations. While both approaches were able to tackle the process family design problem, our newly proposed surrogates-based approach is able to provide solutions with an order of magnitude reduction in the computational time required to produce the necessary input data. We demonstrated these approaches on a transcritical CO_2 process across a family with 56 installations and allowing two unit designs for each of the three common units in the process. For comparison, designing each of these units for each installation uniquely would have required 168 total

unique designs, while our approach required only six in total.

We are currently applying these approaches to an MEA carbon capture facility, motivated by the challenges for mass deployment of such technology (Wilberforce et al., 2019). For future work we hope to extend these formulations to consider parallel common units and process flexibility, variability, and uncertainty.

Disclaimer This project was funded by the United States Department of Energy, National Energy Technology Laboratory, in part, through a site support contract. Neither the United States Government nor any agency thereof, nor any of their employees, nor the support contractor, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Acknowledgements Sandia National Laboratories is managed and operated by NTESS under DOE NNSA contract DE-NA0003525. This effort was partially funded by the U.S. Department of Energy's Institute for the Design of Advanced Energy Systems (IDAES) supported by the Office of Fossil Energy and Carbon Management's Simulation-Based Engineering/Crosscutting Research Program.

References

Anderson, R., J. Huchette, W. Ma, C. Tjandraatmadja, and J. P. Vielma (2020). Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming 183*(1), 3–39.

Arora, A., J. Li, M. S. Zantye, and M. F. Hasan (2020). Design standardization of unit operations for reducing the capital intensity and cost of small-scale chemical processes. *AIChE Journal 66*(2), e16802.

Baldea, M., T. F. Edgar, B. L. Stanley, and A. A. Kiss (2017). Modular manufacturing processes: Status, challenges, and opportunities. *AIChE Journal 63*(10), 4262–4272.

Baud-Lavigne, B., B. Agard, and B. Penz (2016). Simultaneous product family and supply chain design: An optimization approach. *International Journal of Production Economics 174*, 111–118.

Bhosekar, A. and M. G. Ierapetritou (2020). Modular design optimization using machine learning-based flexibility analysis. *Journal of Process Control*.

Bynum, M. L., G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, and D. L. Woodruff (2021). *Pyomo – Optimization Modeling in Python*. Springer.

Ceccon, F., J. Jalving, J. Haddad, A. Thebelt, C. Tsay, C. D. Laird, and R. Misener (2022). Omlt: Optimization & machine learning toolkit. submitted, preprint on arXiv.

Chen, Q. and I. E. Grossmann (2019). Effective generalized disjunctive programming models for modular process synthesis. *Industrial & Engineering Chemistry Research 58*(15), 5873–5886.

Chen, Q., E. S. Johnson, D. E. Bernal, R. Valentin, S. Kale, J. Bates, J. D. Siirola, and I. E. Grossmann (2022). Pyomo. gdp: an ecosystem for logic based modeling and optimization development. *Optimization and Engineering 23*(1), 607–642.

Fischetti, M. and J. Jo (2018). Deep neural networks and mixed integer linear optimization. *Constraints 23*(3), 296–309.

Grimstad, B. and H. Andersson (2019). Relu networks as surrogate models in mixed-integer linear programs. *Computers & Chemical Engineering 131*, 106580.

Lee, A., J. H. Ghouse, J. C. Eslick, C. D. Laird, J. D. Siirola, M. A. Zamarripa, D. Gunter, J. H. Shinn, A. W. Dowling, D. Bhattacharyya, et al. (2021). The idaes process modeling framework and model library—flexibility for process simulation and optimization. *Journal of Advanced Manufacturing and Processing 3*(3), e10095.

Li, D. and E. A. Groll (2005). Transcritical co2 refrigeration cycle with ejector-expansion device. *International Journal of Refrigeration 28*(5), 766–773.

Optimization, LLC Gurobi et al. (2020). Gurobi optimizer reference manual, 2020. *URL http://www. gurobi. com 12*.

Sanchez, R. and T. Shibata (2021). Modularity design rules for architecture development: Theory, implementation, and evidence from the development of the renault–nissan alliance "common module family" architecture. *Journal of Open Innovation: Technology, Market, and Complexity 7*(4), 242.

Simpson, T. W., J. Jiao, Z. Siddique, and K. Hölttä-Otto (2014). Advances in product family and product platform design. *New York: Springer 1*.

Thonemann, U. W. and M. L. Brandeau (2000). Optimal commonality in component design. *Operations Research 48*(1), 1–19.

Towler, G. and R. Sinnott (2022). *Chemical Engineering Design - Principles, Practice and Economics of Plant and Process Design (3rd Edition)*. Elsevier.

Wilberforce, T., A. Baroutaji, B. Soudan, A. H. Al-Alami, and A. G. Olabi (2019). Outlook of carbon capture technology and challenges. *Science of the total environment 657*, 56–72.

Zhang, C., C. Jacobson, Q. Zhang, L. T. Biegler, J. C. Eslick, M. A. Zamarripa, D. Miller, G. Stinchfield, J. D. Siirola, and C. D. Laird (2022). Optimization-based design of product families with common components. In *Computer Aided Chemical Engineering, Proceedings of 14th International Symposium on Process Systems Engineering (PSE 2021+)*, Volume 49, pp. 91–96. Elsevier.