
Article:
Gopinath, S. orcid.org/0009-0004-2872-2440 and Adjiman, C.S. orcid.org/0000-0002-4573-7722 (2025) Superstructure optimization with rigorous models via an exact reformulation. Computers & Chemical Engineering, 194. 108972. ISSN 0098-1354

https://doi.org/10.1016/j.compchemeng.2024.108972

Supporting Information for Superstructure optimization with rigorous models via an exact reformulation

Smitha Gopinath[1] and Claire S. Adjiman[2]

[1]School of Chemical, Materials and Biological Engineering, The University of Sheffield, Mappin Street, Sheffield, S1 3JD, United Kingdom

[2]Imperial College London, Department of Chemical Engineering, Sargent Centre for Process Systems Engineering and Institute for Molecular Science and Engineering, London SW7 2AZ, United Kingdom

March 7, 2025

1 Discussion on master problem initialization- PS and MO environments

In the master problem initialization step of the LBOA algorithm, linearizations of nonlinear conditional constraints across all disjunctions are added to the master problem. In the context of SO, linearizations of constraints corresponding to all the process units considered in the flowsheet are added to the master problem in the initialization step. To do this, one or more primal problems are solved (and subsequently linearized) such that all units are selected at least once. To contrast the initialization strategies in MO and PS formulations

that have been proposed in the literature, we consider the example of distillation column synthesis, in which the optimal number of stages in a column is determined, via state-of-the-art MO (Yeomans and Grossmann, 2000) and hybrid PS-based GDP (Caballero, 2015) formulations. Let N identical equilibrium stages (trays) be allowed in each of the rectifying and stripping sections. In the MO formulation (Yeomans and Grossmann, 2000), $2N$ binary variables are defined, each corresponding to a stage in the column. In the PS formulation (Caballero, 2015), $2N$ binary variables are defined: N of these variables correspond to rectifying column sections such that section i, $i = 1, \ldots, N$ has i stages. Similarly, the next N variables denote N stripping column sections. To initialize the master problem in MO, Yeomans and Grossmann (2000) solve only one primal problem in which all the stages in the column are selected and derive a linearization around its solution. To initialize the PS-based master problem (Caballero, 2015), at least N columns need to be optimized such that each of the rectifying and stripping sections is chosen at least once. Thus, in state-of-the-art PS-based GDP, the number of reduced primal problems to be solved during initialization may grow with the number of discrete alternatives. Further, when only one column section is to be optimized (for e.g., gas-liquid absorption columns), the initialization step entails enumeration of all discrete alternatives. In practice, Caballero (2015) observed that the solution obtained may be sensitive to the choice of primal problems around the solutions of which linearizations are accumulated in the initialization step.

2 Full details of the Implementation of the SO-PS algorithm

The optimization algorithm involves the solution of a continuous primal problem and a mixed-integer master problem. The primal problem, a continuous nonlinear program (NLP) derived by fixing the discrete variables in the MINLP Problem (MSON), is implemented and solved in the modelling environment gPROMS 7.1 (Siemens, 2024). The master problem is a mixed integer linear programming problem which is solved using

Gurobi 10.0.2 (Gurobi Optimization, Inc., 2024) via the C++ API.

The primal problem is solved in gPROMS using a sequential quadratic programming (SQP) solver. In the primal problem, the values of the binary variables and all other discrete variables are fixed *a priori*. The NLP solver takes a feasible path approach with respect to a subset of the SO problem constraints, namely, those equality constraints that are typically solved by the use of a variant of Newton's method. We refer to these constraints as simulator-only constraints and they are treated implicitly by the optimizer. In our implementation the flowsheet-level equality constraints Equation (19b), the process unit-level equality constraints (Equation(19d)), the mixer equality constraints (Equations (19i), (19k)) and the splitter equality constraints (Equations (19j), (19m)) are solved as simulator-only constraints. The remaining constraints in Problem (MSON) are referred to as optimization-only constraints. The variables chosen as degrees of freedom are declared as optimization variables $\boldsymbol{w}$ and the objective function and optimization-only constraints are treated as functions of the optimization variables.

In this work, we treat conditional big-M optimization-only constraints (that are non-linear in the optimization variables) of the following form:

$$-\boldsymbol{M}^u(1-z_u) \leq \boldsymbol{l}^u(\boldsymbol{w}) \leq \boldsymbol{M}^u(1-z_u) \qquad \forall u \in \mathcal{T} \tag{S1}$$

as conditional (on/off) equality constraints, where $\boldsymbol{l}^u$ is a vector of v^u nonlinear conditional constraints that are true when z_u is one and M_i^u is an upper bound on the magnitude of $l_i^u,\ i = 1 \dots v^u$. Examples of such constraints include Equation (16), in which the output variables are implicit functions of the optimization variables. Such an on/off constraint is implemented in the primal problem with the following complementarity constraint:

$$z_u \boldsymbol{l}^u(\boldsymbol{w}) = \boldsymbol{0}, \qquad \forall u \in \mathcal{T} \tag{S2}$$

We reiterate that the value of z_u is fixed during the solution of the primal problem, and hence no additional non-linearity is introduced into the primal problem. Based on the

procedure given in Kocis and Grossmann (1987), we derive equivalent conditional inequality constraints that are added to the master problem,

$$\delta_i^{u(k)}(l_i^u(\boldsymbol{w}^{(k)}) + \nabla_{\boldsymbol{w}} l_i^u(\boldsymbol{w}^{(k)})[\boldsymbol{w} - \boldsymbol{w}^{(k)}]^{\mathrm{T}}) \leq M_i^u(1 - z_u) \qquad i \in \{1, \ldots, v^u\},\ u \in \mathcal{T} \tag{S3}$$

where $\boldsymbol{w}^{(k)}$ is the value of the optimization variables at the solution of the primal problem k; the value of $\delta_i^{u(k)}$ depends on the sign of $\lambda_i^{u(k)}$, the Lagrange multiplier (in accordance with the sign convention used in Kocis and Grossmann (1987)) associated with constraint $z_u l_i^u(\boldsymbol{w}) = 0$ (Equation(S2)) at the solution of primal problem k; $\delta_i^{u(k)} = -1$ when $\lambda_i^{u(k)}$ is strictly negative, $\delta_i^{u(k)}$ when $\lambda_i^{u(k)}$ is strictly positive and zero otherwise; and M is a suitable upper bound on the magnitude of $\boldsymbol{l}(\boldsymbol{w})$. Treating the Big-M constraints (S1) in the form (S2) can avoid convergence issues in the primal problem that may otherwise arise due to the use of these linearly-dependent constraints (as the binary variables are fixed) (Dowling and Biegler, 2015). Further, the approach also avoids linearizing both sides of the nonlinear Big-M inequalities (S1) while constructing the master problem, which would result in an invalid outer-approximation of the feasible region of the constraint (see Kocis and Grossmann (1987) for more details).

As we enforce and linearize conditional constraints only when the associated binary variable is true, similar to the LBOA algorithm, the construction of the master problem in this work is also in accordance with LBOA. In the initialization phase, one or more primal problems are solved such that, across all of the initialization iterations, each process unit is selected at least once. We note that the allowable combinations of selected units are governed by the unit-selection constraints (Equation (19q)) and this in turns determines the the required number of initialization iterations n_I, $1 \leq n_I \leq |\mathcal{T}|$. The initialization phase is followed by the solution phase in which the primal and master problems are solved alternately and the values of the discrete variables for each primal problem $\boldsymbol{z}^{(k)}$ are fixed to the solution of the previous master problem.

At the start of the LB-OA-ER-AP algorithm (logic-based OA-ER-AP), all the opti-

mization constraints that are linear in the degrees of freedom are added to the master problem. After each solution of the primal problem, irrespective of whether the algorithm is in the initialization or solution phase, the following constraints are added to the master problem: i) if the primal problem is feasible, linearizations of the objective function and of the nonlinear optimization constraints and ii) an integer cut to prevent repetition of previously-tested binary values. Due to the inherent non-convexity of Problem (MSON), a slack variable that allows violation of the linearizations is added to each of these and a penalty term is introduced in the objective function of the master problem to minimize these violations (Kocis and Grossmann, 1987). For all constraints we use the same value of the penalty term and set it greater than the magnitude of the largest Lagrange multiplier of all optimization-only constraints across the initialization iterations. The gPROMS gO:RUN functionality is used to solve the primal problem and access the value of the solution, optimization-only constraints and their first derivatives whereas the gPROMS Foreign Object facility is used to pass the assignment of the binary variables $\boldsymbol{z}^{(k)}$ to the primal problem. Unlike several other simulation-based superstructure optimization studies, we do not estimate any first-order derivatives (used to construct the linearizations in the master problem) via finite differences in this work but instead use the derivatives computed by the simulator. We also note that the simulator model (or flowsheet) remains unchanged at each iteration of the algorithm, except for a change in the vector $\boldsymbol{z}^{(k)}$.

We also modify the heuristic stopping criteria of the LB-OA-ER-AP algorithm, akin to Bowskill et al. (2020). The algorithm converges to a solution when whichever of the following occurs: i) in three of the iterations the primal problem is feasible and its objective function is greater than the best known upper bound; ii) the master problem is infeasible and iii) the total number of iterations exceeds 100.

We note that Problem (MSON) can also be formulated directly in gPROMS and solved using the built-in implementation of the OA-ER-AP algorithm and it can be also be formulated in other environments (e.g., GAMS (GAMS Development Corporation, 2024) or PYOMO (Bynum et al., 2021)) and solved using any of the MINLP/GDP algorithms supported.

2.1 Initialization of the simulator-only constraints with MSON

The MSON formulation as implemented with the PS-SO algorithm can be exploited to facilitate the initialization of the simulator-only constraints in the synthesis of complex flowsheets. As mentioned in the implementation section, the mixer-splitter unconditional constraints as well as the constitutive equations of all the process units are all implemented in the simulator. The feasibility of these simulator-only constraints at the initial point is essential as the primal problem is solved via a feasible path approach. However, due to the mixers and splitters which allow flow between units, the equality constraints of all the process units are coupled, and this may make initialization of the simulator-only constraints challenging. To ease initialization of the simulator-only constraints arising from Problem (MSON), here we create Problem (MSON-i), a copy of Problem (MSON) in which the following alterations are introduced to the mixer-splitter network:

- Each mixer associated with unit $u \in \mathcal{U}\backslash\mathcal{T}$ is replaced by a modified mixer. Suitable values for f_i^{A}, $\boldsymbol{q}_i^{\mathrm{A}}$ and T_i^{A}, $i \in \mathcal{IN}_u$, $u \in \mathcal{U}\backslash\mathcal{T}$ are identified.

- Each splitter associated with unit $u \in \mathcal{U}\backslash\mathcal{T}$ is replaced by a modified splitter.

The degrees of freedom of the simulator-only constraints arising from Problem (MSON-i), which also include the flowrates between units, are next initialized in the following manner:

1. $f_{o,i} := 0$ if o is a splitter associated with any unit $u \in \mathcal{F} \cup \mathcal{L}$ and i is a mixer associated with any unit $v \in \mathcal{L} \cup \mathcal{P}$.

2. $f_i^{\mathrm{M}} := f_i^{\mathrm{A}}$ for all units $u \in \mathcal{L} \cup \mathcal{P}$.

3. $\boldsymbol{x}_{\mathcal{A}_u} := \boldsymbol{x}_{\mathcal{A}_u}^{\mathrm{A}}$ and $\boldsymbol{y}_{\mathcal{A}_u} := \boldsymbol{y}_{\mathcal{A}_u}^{\mathrm{A}}$ for all units $u \in \mathcal{L}$.

With the proposed initial setting of all the flowrates between units to zero, the feeds into the superstructure bypass the process units and flow out directly via the artificial outlet streams in the modified splitters associated with the source units. This assignment of the

flowrates between units thus decouples all the units, that is, no mass flows from any unit $u \in \mathcal{F} \cup \mathcal{L}$ into any other unit $u \in \mathcal{L} \cup \mathcal{P}$. Additionally, for each unit $u \in \mathcal{L}$, the MESH equations are solved without difficulty due to the artificial mass flows from the modified mixers that are associated with the unit, as well as a consistent assignment of the degrees of freedom that correspond to the unit only. The modified mixers and splitters enable the decoupling of process units and thus result in the solution of the simulator-only constraints corresponding to Problem (MSON-i) at the initial point given above.

Further, Problem (MSON-i) is fully equivalent to Problem (MSON) when the following optimization-only constraints are added to its formulation:

- $f_i^{\mathrm{M}} = 0$ if i is a mixer associated with any unit $u \in \mathcal{P} \cup \mathcal{L} \setminus \mathcal{T}$.
- $f_o^{\mathrm{S}} = 0$ if o is a splitter associated with any unit $u \in \mathcal{F} \cup \mathcal{L} \setminus \mathcal{T}$.

The above constraints enforce that the modified mixers and splitters that are associated with permanent units behave like standard mixers and splitters as the flowrates of the ficitious streams are set to zero. This equivalent formulation of Problem (MSON) enables the systematic and robust initialization of superstructure optimization problems.

3 Initialization of the counter-current column in PS environments

We find that the modified mixers and splitters provide a simple route to initialization of the process unit-level constraints that are implemented in the process simulator. We first decouple column subsections by using the initialization procedure in section 2.1. For each of the fictitious streams in the modified liquid and vapour mixers, we set the flowrates, $f^{\mathrm{A,L}}$ and $f^{\mathrm{A,V}}$, and compositions, $\boldsymbol{q}^{\mathrm{A,L}}$ and $\boldsymbol{q}^{\mathrm{A,V}}$ and temperatures, $T^{\mathrm{A,L}}$ and $T^{\mathrm{A,V}}$, equal to that of a liquid and a vapour that are in equilibrium with each other, inspired by a counter-current column initialization procedure given in Keskes (2007). It follows that in this case, the liquid (vapour) streams at the outlet of each stage and each column subsection are

identical to the streams at the inlets. Thus, a feasible solution to the MESH equations of each equilibrium stage as well as each column subsection in the column superstructure may easily be determined. The generated solution is then be used to initialize a simple counter-current column with strictly positive flows between the process units.

References

Bowskill, D. H., Tropp, U. E., Gopinath, S., Jackson, G., Galindo, A., and Adjiman, C. S. (2020). Beyond a heuristic analysis: integration of process and working-fluid design for organic rankine cycles. *Molecular Systems Design & Engineering*, 5:493–510.

Bynum, M. L., Hackebeil, G. A., Hart, W. E., Laird, C. D., Nicholson, B. L., Siirola, J. D., Watson, J.-P., and Woodruff, D. L. (2021). *Pyomo Overview*, pages 25–36. Springer International Publishing, Cham.

Caballero, J. A. (2015). Logic hybrid simulation-optimization algorithm for distillation design. *Computers & Chemical Engineering*, 72:284 – 299.

Dowling, A. W. and Biegler, L. T. (2015). Degeneracy hunter: An algorithm for determining irreducible sets of degenerate constraints in mathematical programs. In Gernaey, K. V., Huusom, J. K., and Gani, R., editors, *12th International Symposium on Process Systems Engineering and 25th European Symposium on Computer Aided Process Engineering*, volume 37 of *Computer Aided Chemical Engineering*, pages 809–814. Elsevier.

GAMS Development Corporation (2024). General Algebraic Modeling System (GAMS) release 36.1.0, Fairfax, VA, USA, 2021. https://www.gams.com/download/.

Gurobi Optimization, Inc. (2024). Gurobi optimizer reference manual. http://www.gurobi.com.

Keskes, E. (2007). *Integrated process and solvent design for C0 2 removal from natural gas.* PhD thesis, Imperial College London.

Kocis, G. R. and Grossmann, I. E. (1987). Relaxation strategy for the structural optimization of process flow sheets. *Industrial & Engineering Chemistry Research*, 26(9):1869–1880.

Siemens (1996-2024). gPROMS. https://www.siemens.com/global/en/products/automation/industry-software/gproms-digital-process-design-and-operations.

Yeomans, H. and Grossmann, I. E. (2000). Disjunctive programming models for the optimal design of distillation columns and separation sequences. *Industrial & Engineering Chemistry Research*, 39(6):1637–1648.