Flavio Manenti, Gintaras V. Reklaitis (Eds.), Proceedings of the 34th European Symposium on Computer Aided Process Engineering / 15th International Symposium on Process Systems Engineering (ESCAPE34/PSE24), June 2-6, 2024, Florence, Italy
 http://dx.doi.org/10.1016/B978-0-443-28824-1.50214-3

# Hybrid Artificial Intelligence-based Process Flowsheet Synthesis and Design using Extended SFILES Representation

Vipul Mann,[a] Mauricio Sales-Cruz[b], Rafiqul Gani,[c,d,e] Venkat Venkatasubramanian[a,*]

[a]*Columbia University, New York 10027, USA*
[b]*Universidad Autónoma Metropolitana-Cuajimalpa, Ciudad de México, México*
[c]*PSE for SPEED Company, DK-2920, Charlottenlund, Denmark*
[d]*The Hong Kong University of Science and Technology (Guangzhou), China*
[e]*Széchenyi István University, 9026 Győr, Hungary*

## Abstract

Process flowsheet synthesis and design involves simultaneously solving several problems, including determining the unit operations and their sequence, underlying reactions and reaction stoichiometry, downstream separation design and operation parameters, sustainability factors, and many more. Naturally, this results in a large amount of data being associated with a given process flowsheet that captures the relevant process context and should be readily accessible. This data is useful for solving related problems both using data-driven and process knowledge-based methods. A hierarchical framework, called the extended SFILES (or eSFILES), proposed recently stores this information using a combination of text-based, graph-based, and ontology-based representations. Here, we provide details on a prototype software for automated flowsheet representation and generation across various levels in the eSFILES framework. The underlying methods include a novel flowsheet grammar, a set of inferencing algorithms, and interfacing with a commercial process simulator facilitating rigorous flowsheet simulation.

**Keywords**: process design, flowsheet modeling, artificial intelligence, computer-aided flowsheet synthesis

## 1. Introduction

A central problem in process systems engineering is to efficiently convert raw materials to desired products, which involves evaluating the correct sequence of unit operations, their design, and optimization of associated operations, also known as the process synthesis and design problem. The problem becomes computationally complex due to the various decisions to be made at each stage. To mitigate the challenges associated with this, we recently reported a multi-level flowsheet representation and generation framework, called extended SFILES (or eSFILES), that could be used to efficiently solve flowsheet synthesis and design problems using hybrid artificial intelligence (AI) methods (Mann et al., 2024), (Mann et al., 2023b). The eSFILES framework represents flowsheet information at varying granularity using three levels with a base level 0. At level 0, flow diagrams are represented as purely text-based SFILES strings (Bommareddy et al., 2011), (D'Anterroches, 2005), (Tula et al., 2015). At level 1, SFILES grammar and inferencing algorithms, are used to construct a flowsheet hypergraph explicitly representing flow-

diagram connectivity. At level 2, specifications needed for material and energy balance calculations are introduced, and their simulation results with simple models are also added using annotated flowsheet hypergraphs. Finally, at level 3, a process ontology is connected with the annotated flowsheet hypergraph to include design and operation parameters and simulation results with rigorous models.

In this work, we present details on an automated tool for representing process flowsheets using the eSFILES representation. The developed parser (software tool) comprises an extensive set of inference algorithms that allow for the conversion of flowsheet representations at a given level to any other level in the framework. Moreover, it also interfaces with commercial process simulators and process generators. Algorithms underlying the parser include – a novel flowsheet grammar that formally defines syntax rules, flowsheet hypergraph generation and inference algorithms, text-based flowsheet representation generation, rigorous process simulation with integration with commercial process simulators, and so on. The prototype program enables the development of hybrid AI systems that efficiently blend domain knowledge with data-driven techniques (Mann et al., 2023a), (Venkatasubramanian & Mann, 2022). We envision the parser would accelerate the development of fast, efficient, and consistent solutions for flowsheet-related problems, that result in hybrid AI systems as opposed to purely data-driven approaches (Vogel et al., 2022) (Hirtreiter et al., 2022).

## 2. Extended SFILES representation

The extended SFILES (eSFILES) representation involves representing flowsheets across four levels (levels 0, 1, 2, and 3) with varying degrees of process knowledge. At the core of the eSFILES representation lies the concept of process-atoms and process-bonds from the SFILES representation, the concept of annotated hypergraphs with process-atoms represented as hyperedges and process-bonds represented as nodes, and a novel flowsheet grammar that is combined with inferencing algorithms to facilitate automated parsing and interconversion across multiple representation levels. To illustrate the multi-level eSFILES framework, consider a simple four-component separation task as shown in Figure 1a where four components A, B, C, and D enter the first distillation column and they need to be separated into pure component streams. Details of other more complex examples can be obtained from the authors.

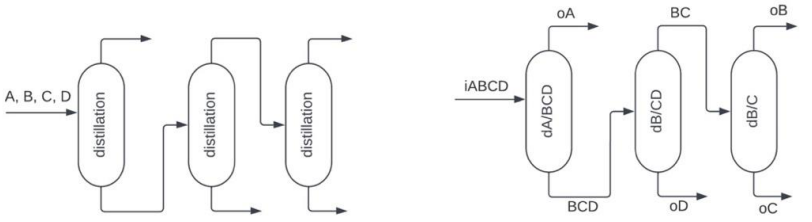


Figure 1. (a) Process flow diagram (b) Process-atoms and process-bonds

*2.1. Level 0: Text-based SFILES representation*

The process-atoms and process-bonds corresponding to the given process flow diagram are shown in Figure 1b with dA/BCD, dB/CD, and dB/C representing the three distillation columns with top and bottom streams separated by '/'. The process-bonds are the raw input stream iABCD, output streams oA, oC, oB, and oC, and the intermediate streams are BCD and BC. The purely text-based representation comprising eSFILES level 0 representation is given by,

(iABCD)(dA/BCD)(oA)(dBC/D)(oD)(dB/C)(oB)(oC)

*2.2. Level 1: Hypergraph-based representation*

Level 1 represents the connectivity between process-atoms explicitly using a flowsheet hypergraph with process-bonds (ABCD, A, BCD, BC, D, B, C) as nodes and process-atoms (D1, D2, D3) as hyperedges as shown in Figure 2a. The graph-based representation is numeric in nature and could be easily combined with graph-theoretic methods to perform superstructure optimization, flowsheet enumeration, and so on.

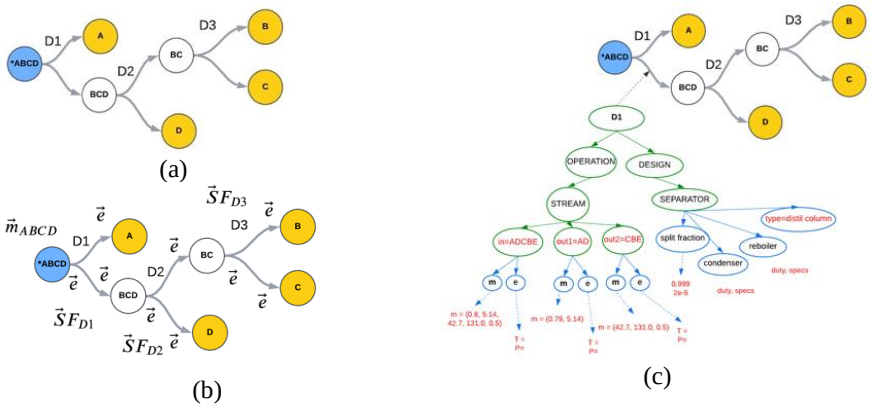


Figure 2. (a) level 1 representation (b) level 2 representation (c) level 3 representation

*2.3. Level 2: Annotated flowsheet hypergraph*

Level 2 represents additional flowsheet context such as material and energy balance information using node-edge paired annotations in the hypergraph, thus allowing for multiple annotations for the same node connected to different hyperedges. These annotations include component flow rate (s) $m_i$ for streams, split fraction $SF_j$ for each separator, and temperature $T$ and pressure $P$ for each stream entering or leaving the process-atom, as shown in Figure 2b. This representation facilitates the use of simple models for material and energy balance calculations.

*2.4. Level 3: Process ontology-connected flowsheet hypergraph*

Level 3 represents information on design and operation parameters using a custom-built process ontology connected to the hypergraph via hyperedges (see Figure 2c). Each hyperedge (process-atom) is connected to the instantiated process ontology where the relevant design and operation information are organized hierarchically as classes, subclasses, and properties with process-atom as the main class. This representation is used for rigorous flowsheet calculations using commercial process simulators.

For further details on various levels of the eSFILES framework with additional examples of reactors, recycle streams, and so on, the reader is referred to (Mann et al., 2024).

## 3. Automatically parsing and generation of eSFILES representation

The primary objective of the developed flowsheet parser is the conversion of the base level, i.e., level 0 of the flowsheet representation to level 1 automatically, and further generation of higher-level representations (levels 2 and beyond) by interfacing with commercial process simulators. This requires inferring the underlying process-atoms, process-bonds, recycle streams, sequence and connectivity patterns, and the input/output

streams in the process. While the level 0 string representation, in principle, contains this information, a formal set of rules, heuristics, and algorithms are required to achieve this automation, namely, flowsheet grammar and a set of inferencing algorithms described in the following sections.

### *3.1. Flowsheet grammar*

The flowsheet grammar defines the set of rules that formalize the structure of the text-based representation and is characterized by a set of symbols and recursive rules that could be applied systematically to generate grammatically valid SFILES strings. Formally, it consists of – S, a designated start symbol; $\sum$, the set of terminal symbols; N, the set of non-terminal symbols; and R, the set of syntax rules of the form $A \rightarrow \beta$, where A is a non-terminal symbol and $\beta$ could either be a terminal symbol (from $\sum$) or a non-terminal symbol (from N). A subset of the developed grammar rules is listed in Table 1. The corresponding grammar symbols are – S: SFILES, $N$: {i,A,B,C,D}, $\sum$: {PA, PG, EQUIPMENT, INLET, OUTLET, SEP, …}, and $R$: $R_{1-20}$. The hierarchical grammar tree corresponding to the SFILES string for the four-component separation is shown in Figure 3 below. For a complete list of the developed flowsheet grammar and additional examples of grammar trees, please refer to (Mann et al., 2024).

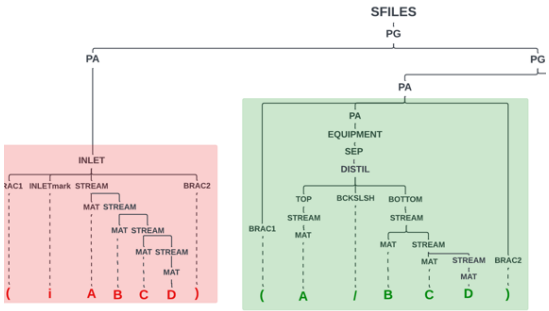


Figure 3: Partially shown grammar syntax tree with inlet and distillation hierarchy

Table 1: A subset of the developed SFILES grammar rules

| Rule | Grammar syntax rules |
|---|---|
| $R_1$ | SFILES → PG |
| $R_2$ | PG → PA PG \| PA |
| $R_4$ | PA → BRAC1 PA BRAC2 |
| $R_6$ | PA → INLET \| OUTLET \| EQUIPMENT |
| $R_8$ | INLET → BRAC1 INLETmark STREAM BRAC2 |
| $R_9$ | OUTLET → BRAC1 OUTLETmark STREAM BRAC2 |
| $R_{10}$ | EQUIPMENT → SEP |
| $R_{11}$ | SEP → DISTIL |
| $R_{12}$ | DISTIL → TOP BCKSLSH BOTTOM |
| $R_{13}$ | TOP \| BOTTOM → STREAM |
| $R_{14}$ | STREAM → STREAM MAT \| MAT |
| $R_{15}$ | MAT → A \| B \| C \| D |
| $R_{16}$ | BRAC1 → ( |
| $R_{17}$ | BRAC2 → ) |
| $R_{18}$ | BCKSLSH → / |
| $R_{19}$ | INLETmark → i |
| $R_{20}$ | OUTLETmark → o |

### *3.2. Inferencing algorithms*

#### *3.2.1. Grammar parsing*

Given a SFILES string for a process flowsheet, a hierarchical grammar syntax tree indicating additional structural information on the flowsheet string is generated as shown in Figure 3. The grammar rules are generated by providing the complete set of grammar rules (similar to those given in Table 1) to natural language parsers like ChartParser in the NLTK library in Python. The ChartParser uses dynamic programming to efficiently parse a given string and generates a tree structure for each string subsequence using the provided set of grammar syntax rules. For invalid strings, the grammar tree would not be generated, resulting in a parsing error. Thus, the generation of such hierarchical trees facilitate automated syntax checking for SFILES strings.

#### *3.2.2. Inferring process-atoms and process-bonds*

In the grammar tree, all the process-atoms and process-bonds are identified by extracting subtrees from the parent tree by filtering them on intermediate node labels. For instance,

the inlet, outlet, and recycle streams are identified by extracting subtrees with labels 'INLET', 'OUTLET', and 'RECYCLE'; for intermediate streams, subtrees with label 'STREAM' AND parent labels not in ['INLET', 'OUTLET', 'RECYCLE'] are extracted; the individual materials are identified as subtrees with parent node 'MAT'; separators correspond to subtrees with label 'SEP' and their type is the immediate child node label such as 'DISTIL'; the top and bottom streams, reactor reactant and product streams, and so on are identified based on similar grammar tree-based inferencing.

*3.2.3. Inferring connectivity*

Connectivity is inferred based on tree-distance between subtrees identified for each process-atom. For computing the distance, the root node for each subtree is represented as a sequence of integers based on whether a given node is on the left or right of the parent node, with an additional integer for each level of depth. For instance, the position of subtree with node 'SEP' in Figure 3 would be represented as (0,1,0,1,0,0). The grammar structure and the SFILES string structure by design is such that adjacent process-atoms are connected together linearly. Each process-atom would have a left-connectivity and right-connectivity, unless they are input or output streams that just have right-connectivity and left-connectivity, respectively. The nearest left and right subtrees are identified based on tree distances, to identify the left connection and right connection of a given process-atom. Since we have information on the top/bottom or reactant/product streams from the grammar tree, the output of the left process-atom automatically becomes one of the inputs of the current process-atom. This also allows performing material balance-based sanity checks while inferring connectivity.

*3.2.4. Additional information for level 2*

For levels 2 and 3, additional information needs to be input by the user or generated using intelligent design methods similar to those proposed in Tula et al. (2015). For level 2, the information on material and energy balance calculations is required. This information could be obtained from flowsheet generators. The input to such flowsheet generators is the level 1 flowsheet hypergraph and the process context (design goals, etc.) and the output is the information required for simple material and energy balance calculations stored as node-edge labels as shown in Figure 2b. The level 2 representations are used for performing mass and energy balance calculations using simple models.

*3.2.5. Additional information for level 3*

For level 3, additional information such as the design and operation parameters are required to perform process flowsheet simulations. This information is obtained from commercial process simulators and stored as an instantiated ontology connected to the process-atom hyperedge as shown in Figure 2c. The connection between the eSFILES framework and process simulators is enabled by using the keyword file to translate the possibly different nomenclature between the two systems. Such keyword files have a unique variable name for each piece of information required to simulate the process, and thus, allow information flow between level 3 representation and process simulator input and output. For instance, a snippet of the interfacing between eSFILES level 3 and process simulators such as Pro II is shown in Figure 4. These keywords are mapped to our developed process ontology and connected to the flowsheet hypergraph, facilitating information exchange between the two systems required for rigorous simulations.

## 4. Conclusions

An automated software tool for performing hybrid AI-based process flowsheet synthesis and design using a multi-level hierarchical flowsheet representation and generation

framework called extended SFILES (or eSFILES) has been presented. The developed software tool allows automated interconversion of flowsheet representations at various levels in the eSFILES framework with varying degrees of process flowsheet information. The level 0 (base level) text-based flowsheet representation is converted to level 1 flowsheet hypergraph using a flowsheet grammar combined with tree-based inferencing algorithms to infer unit operations, streams, and connectivity between them. The level 2 and level 3 representations are generated using the level 1 representation and interfacing with process simulators and generators using keyword files. A custom-built process ontology map stores the information from keyword files and is connected to the hypergraph containing relevant information for each process-atom, thus facilitating information sharing between the two systems. The development of the prototype software would aid in the further development of hybrid AI systems that not only leverage process knowledge-based methods but also take advantage of the computational benefits of data-driven methods, thereby allowing reliable, efficient, and rapid process development.

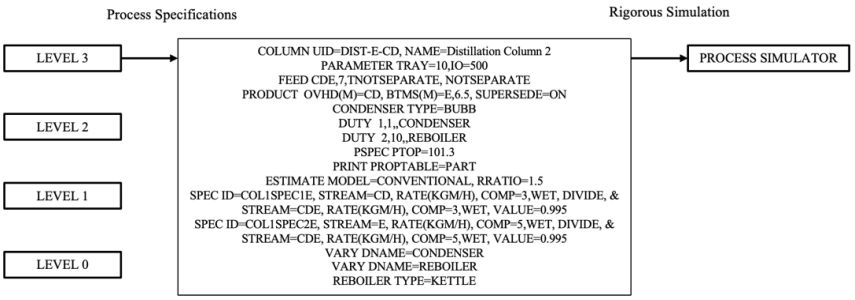


Figure 4: Interfacing between level 3 and commercial process simulators

## References

Bommareddy, S., Eden, M. R., & Gani, R. (2011). *Computer Aided Flowsheet Design using Group Contribution Methods* (pp. 321–325). https://doi.org/10.1016/B978-0-444-53711-9.50065-1

D'Anterroches, L. (2005). *Process flowsheet generation & design through a group contribution approach*. [CAPEC], Department of Chemical Engineering, Technical University of Denmark.

Hirtreiter, E., Balhorn, L. S., & Schweidtmann, A. M. (2022). *Towards automatic generation of Piping and Instrumentation Diagrams (P&IDs) with Artificial Intelligence*. http://arxiv.org/abs/2211.05583

Mann, V., Gani, R., & Venkatasubramanian, V. (2023a). Group contribution-based property modeling for chemical product design: A perspective in the AI era. *Fluid Phase Equilibria*, *568*. https://doi.org/10.1016/j.fluid.2023.113734

Mann, V., Gani, R., & Venkatasubramanian, V. (2023b). Intelligent Process Flowsheet Synthesis and Design using Extended SFILES Representation. In *Computer Aided Chemical Engineering* (Vol. 52, pp. 221–226). Elsevier B.V. https://doi.org/10.1016/B978-0-443-15274-0.50036-6

Mann, V., Sales-Cruz, M., Gani, R., & Venkatasubramanian, V. (2024). eSFILES: Intelligent process flowsheet synthesis using process knowledge, symbolic AI, and machine learning. *Computers & Chemical Engineering*, *181*, 108505. https://doi.org/10.1016/j.compchemeng.2023.108505

Tula, A. K., Eden, M. R., & Gani, R. (2015). Process synthesis, design and analysis using a process-group contribution method. *Computers & Chemical Engineering*, *81*, 245–259. https://doi.org/10.1016/J.COMPCHEMENG.2015.04.019

Venkatasubramanian, V., & Mann, V. (2022). Artificial intelligence in reaction prediction and chemical synthesis. *Current Opinion in Chemical Engineering*, *36*, 100749. https://doi.org/10.1016/j.coche.2021.100749

Vogel, G., Balhorn, L. S., & Schweidtmann, A. M. (2022). *Learning from flowsheets: A generative transformer model for autocompletion of flowsheets*. http://arxiv.org/abs/2208.00859