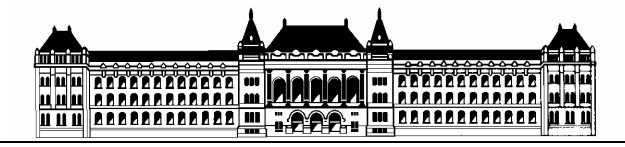


BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
Department of Chemical Engineering

Chemical Process Synthesis Using Mixed Integer Nonlinear Programming

A thesis submitted to the

Budapest University of Technology and Economics

for the degree of

Doctor of Philosophy

by

Tivadar Farkas

under the supervision of

Zoltán Lelkes, Ph.D.
associate professor

2006

Acknowledgments

First of all, I would like to thank my supervisor, associate professor **Zoltán Lelkes**, for his guidance**,** and for his support during my time at the Chemical Engineering Department at the Budapest University of Technology and Economics.

I am also very grateful to associate professor **Endre Rév**, for his continuous assistance and for readily helping me during my PhD studies. Their suggestions greatly improved the level of the research I have done.

I would like to express my acknowledgment to professor **Zsolt Fonyó** , who, to our great regret, is not in our midst.

I would like to thank professor **Andzrej Kraslawski** (Department of Chemical Technology, Lappeenranta University of Technology, Finland), and professor **Zdravko Kravanja** (Faculty of Chemistry and Chemical Engineering, University of Maribor, Slovenia) that they shared their great expertise with me.

Herewith I would like to thank **all the colleagues at the Chemical Engineering Department at the Budapest University of Technology and Economics**, especially to **Mrs. Gabriella Ling-Mihalovics**, for maintaining a friendly and family atmosphere at the department.

I am really grateful to **all my friends**, who raised my spirit, tugged me away from the desk, and cheered me up whenever it was necessary, and whose names unfortunately cannot be listed here, because the dissertation should not be twice long as now.

I am forever indebted to **my parents** and to **my sister** that they encouraged and supported my studies from the very beginning of my life.

Table of Contents

Introduction

The target of process synthesis is to discover the best complete design to accomplish a chemical-manufacturing goal. In this process first the alternatives have to be considered. Usually most of the alternatives can be ruled out according to engineering experience, but even the number of remained alternatives is even so huge, that process synthesis needs systematic study.

The term 'synthesis' was introduced in the late-1960s (Masso and Rudd, 1969), and the first review article was published in 1973 (Hendry et al.). Since then several papers dealt with the searching methods, the representations of flowsheets, and the objective functions applied in different branches of the chemical engineering.

The searching methods can be classified into three groups: targeting methods, knowledge based methods and optimisation techniques. Targeting methods determine certain features that a „good", near optimal design should exhibit. For example using the pinch analysis of Linnhoff (1993, 1994) the minimum utility assumptions in a heat exchanger network can be determined without any design. However, after the use of targeting methods the engineer usually have to do the design using another technique. Knowledge based methods are based on the engineering experience, or the rules of thumb. Maybe the heuristic evolutionary method of Douglas (1985, 1988) is the most known method among them. In this method the original problem is decomposed into five simpler levels, and these levels are solved in sequence, based on the engineering experience. The knowledge based methods can find a very good, near optimal solution, but they are fallible. Without earlier experience, or the collection of earlier solved problems, it is very difficult to use them.

In the optimisation techniques a mathematical representation of the problem is generated, and optimised. These approaches have two main groups: stochastic and deterministic optimisation methods. Stochastic optimisation methods can deal with huge, complex problem, and can handle any degree of nonlinearities or discontinuities. These approaches can find near optimal solution very fast; however, they cannot guarantee the global optimality of the solution. Deterministic optimisation methods, i.e. mathematical programming, perform the most rigorous search. In the last decades these methods have attracted more and more attention because of the fast development of computers and of the increase of computational capacity. They guarantee the global optimum in case of convex problem.

Mathematical programming has three main steps. (1) First a superstructure containing all the considered alternatives, and its graph representation, are generated. (2) Then a mathematical

representation is formulated, based on the graph representation of the superstructure. (3) Finally, the mathematical model is optimised.

The most common mathematical representation, which can handle also discrete decisions, is the mixed integer nonlinear programming (MINLP). Several MINLP models and representations have been published in different branches of chemical engineering (heat exchange networks, mass exchange networks, rectification columns and distillation sequences, reactive distillation, etc.). These models are formulated in such ways that they can be solved easily using one of the optimisation algorithms. Grossmann (1996) defined the three major guidelines of a „good" MINLP representation: (1) Keep the problem as linear as possible. (2) Develop a formulation whose NLP relaxation is as tight as possible. (3) If possible, reformulate the MINLP as a convex programming problem.

Generation of the superstructure is an important part of the synthesis. If the superstructure is not defined properly, the problem can be infeasible, or the real optimum can be excluded from the representation. This step requires engineering experience. Until now only one automatic superstructure generation method is published, by Friedler and co-workers (1992ab, 1993, 1998). However, this method requires engineering considerations, as well, to choose the units from which the superstructure is generated. It seems evident that in generating the superstructure the memory of earlier solved problems and cases should be utilized, for example using a knowledge based method.

Multiplicity causes serious problems in MINLP models. Multiplicity means that several solutions of the mathematical representation define the same structure. It can also be said that a structure is represented by isomorphic graphs. In this case, the objective function has the same value in several different points. It makes more difficult finding the optimum, and the search space is unnecessarily big. Multiplicity is usually decreased in the second step of the mathematical programming, and the MINLP model is formulated in the way as to have as low multiplicity as possible. In some cases, however, even the graph representation of the superstructure can be generated in the way as to exclude isomorphic graphs.

An important characteristic of an MINLP model is the number of its binary variables. With increasing number of binary variables, the complexity of the problem and the solution time increase exponentially. An MINLP model can be reformulated in a way as to decrease the number of binary variables by introducing new continuous variables and constraints. Taking into account the possibility of decreasing the number of binary variables already in the generation of superstructure and graph representation would also be useful.

Mainly the second and the third step of the mathematical programming are studied in the literature. It is not rare that a new MINLP representation is published with a new, most appropriate algorithm. The new algorithms are developed in order to be able to solve larger, more complex problems, to easier handle discontinuities, and nonlinearities. However, sometimes these algorithms can be used only in a very narrow branch of problems. The MINLP representations are usually generated in such a way as to be suitable for solving by a certain algorithm.

In my PhD dissertation I present my results in these topics. In the next chapter the main methods of process synthesis are reviewed. The mathematical programming and the use of MINLP representations in the chemical engineering are detailed. In the third chapter I present the use of a knowledge based method, case-based reasoning, in the selection of proper superstructure with MINLP model in distillation column synthesis. In the fourth chapter I study the relation between structures, graphs, and representations, and I give guidelines for the generation of the superstructure and the MINLP model in order to enhance the possibility of finding the global optimum of the problem. Finally, in the fifth chapter I demonstrate the use of these guidelines in the generation of a new superstructure and MINLP model in distillation column synthesis.

1. Literature review

The goal of process synthesis is to find the optimal process flowsheet according to a given objective function, which is commonly economic in nature. First all the alternatives have to be considered in an implicit or explicit way. Then the optimal structure, i.e. the units, the interconnections among them, and their operational parameters, has to be determined by a searching algorithm. The solution is optimal if the objective reached its extreme, e.g. the total cost is minimal, or the net present value is maximal.
Several reviews have been published about process synthesis. Some of them are: Hendry et al., 1973; Hlavacek, 1978; Nishida et al., 1981; Coulson et al., 1985; Douglas, 1988; Smith, 1995; Ullmann, 1996; Biegler et al., 1997.

Calculating and comparing all the alternatives and choosing the best solution among them seems to be an evident methodology for process synthesis, but usually it is impossible, as it is shown by a simple assignment problem (The New Encyclopædia Britannica, 1990-1999, 'optimisation' entry). The target is to assign 70 jobs to 70 differently qualified workers of a company in a way that all the workers get the best suited work to their qualification. The number of all alternatives is 70! (=70·69·…·2·1). It means that 70! alternatives should be studied; this is about 10^{100}. If this calculation is performed with a computer which studies an alternative in one second then the solution of the problem takes more than 10^{87} years, much more than the estimated age of the universe.
In a complex chemical process the number of possible alternatives can be infinite. Most of these alternatives can be ruled out according to engineering considerations, but even the number of the remaining alternatives is so huge that the process synthesis needs systematic study.

1.1. Searching methods

Grossmann and Daichendt (1996) classified the searching methodologies into three groups: targeting methods, heuristics, and optimisation techniques. Li and Kraslawski (2004) listed other techniques next to heuristics, and called them knowledge based methods.

1.1.1. Targeting methods

Targeting methods determine certain features that a „good", near optimal design should exhibit.

The most known targeting method is perhaps the *pinch analysis* developed by Linnhoff (1993, 1994). It is used for determining the design targets of heat exchanger network (HEN) synthesis, such as minimum cold and hot utility assumptions, using physical and graphical insights. According to the characteristics of the streams, the cold and hot composite curves are drawn, and after finding the pinch point by shifting the composite curves, the minimum cold and hot utility assumptions can be read from the diagram.

El-Halwagi and Manouthiousakis (1989) applied the pinch analysis in mass exchange network (MEN) synthesis based on the analogy between HENs and MENs.

Hallale (1998) developed a *supertargeting* method which determines the target optimal cost of a mass exchange network without any synthesis.

The main advantage of targeting methods is that they provide guidelines without performing any complex design. Their main weakness is that the engineer usually still has to do the design work using an other method.

1.1.2. Knowledge based methods

Knowledge based methods concentrate on the representation and knowledge organisation of the design problem. Usually they use the earlier experience of the engineer, or the search is based on previously solved problems.

The most known knowledge based method is the *heuristic approach* which is based on the long-term experience of engineers, and uses the (usually unproven) rules of thumb in the design. Masso and Rudd (Rudd, 1969; Masso and Rudd, 1969) were among the first to propose using heuristics. They used heuristic rules in the selection of the next match in the sequential synthesis of heat exchanger networks. Douglas (1985, 1988) developed a general *hierarchical decomposition method* for the synthesis of chemical processes. This method breaks down the complex problem into more manageable simpler subproblems. Five decision levels are determined, which are solved in sequence: (1) batch versus continuous; (2) input-output structure of the flowsheet; (3) recycle structure of the flowsheet; (4) separation system synthesis; (5) heat recovery network. The main limitation of this method, due to its sequential nature, is the impossibility to manage the interactions between different design levels. Heuristic approaches in general offer no guarantee of finding the best possible design.

Siirola (1996) applied the *means-ends analysis* in the chemical process synthesis. This is an operation-based state transformation paradigm. In case of a problem the raw materials are considered as initial state, and the goal products as the goal state. If the value of a property of the initial state (e.g. identity, amount, concentration, phase, temperature, pressure) is different from the corresponding property of the goal state, a property difference is detected. The purpose of the method is to apply technologies in systematic sequence such that these property differences are eliminated so that the raw materials become transformed into the desired products. Such differences are reduced or eliminated by using the well known technologies for appropriate properties, such as chemical reaction to change molecular identity, mixing and splitting to change amount, separation to change concentration and purity, enthalpy modification to change phase, temperature, pressure, etc.

Phenomena-driven design proposes that reasoning should not start at the level of building blocks but at a low level of aggregation, i.e. at the level of the phenomena that occur in those building blocks. Jaksland et al. (1995) developed a separation process design and synthesis method based on thermodynamic phenomena. They explored the relationship between the physicochemical properties, separation techniques, and conditions of operations. The method includes a systematic analysis of a wide range of physical and chemical properties of the components of the mixture to be separated. According to this analysis, a binary ratio matrix is computed which represents the property differences between all binary pairs. Then the feasible separation techniques are determined for each binary pair of components taking into account the binary ratio matrix and a matrix of allowable values for the property values. The feasible alternatives are screened, and the split factors are estimated. Finally the separation tasks are sequenced and the conditions of operations are determined.

Case-based reasoning (CBR) imitates a human reasoning and tries to solve new problems reusing solutions that were applied to past similar problems. CBR deals with very specific data from the previous situations, and reuses results and experience to fit a new problem situation (Watson, 1997). In case-based reasoning first the most similar case to the actual problem is retrieved from a case library. If the solution of this most similar case cannot be used for the actual problem, than the earlier solution has to be adapted according to the actual requirements. If the problem is solved then, in the last step, it is incorporated in the case library. Pajula and co-workers (2001) developed a case-based reasoning method for the selection of single separations. Seuranen and co-workers (2005) further developed this approach for the synthesis of complex separation sequences.

Sauar et al. (1996) have proposed a new principle of process design based on the *equipartition of the driving forces*. They claimed that process design should be optimised by the equal distribution of the driving forces throughout the process by assuming that the rates of entropy production are proportional to the square of the driving forces.

d'Anterroches and Gani (2005) developed a *group contribution method* for process synthesis based on the group contribution method for pure component property prediction. In this latter method a molecule is described as a set of groups linked together to form a molecular structure. In the same way, for flowsheet "property" prediction, a flowsheet can be described as a set of process-groups linked together to represent the flowsheet structure. Just as a functional group is a collection of atoms, a process-group is a collection of operations forming a "unit" operation or a set of "unit" operations. The links between the process-groups are the streams similar to the bonds that are attachments to atoms/groups. Each process-group provides a contribution to the "property" of the flowsheet, which can be performance in terms of energy consumption, thereby allowing a flowsheet "property" to be calculated, once it is described by the groups.

Knowledge based methods in general need the use of earlier engineering experience, or an organised collection of earlier solved problems. Based on these principles even very large, or complex problems can be solved. However, without such experience, or solved cases, a new problem can hardly be solved. An other disadvantage of these methods is that they cannot guarantee that the best solution is found. Although they often lead to good, near optimal design, they are fallible.

1.1.3. Optimisation techniques

Optimisation techniques use a formal, mathematical, representation of the problem, and they search the solution by mathematical optimisation. These methods can be classed into two main groups: stochastic (i.e. non-deterministic) and deterministic methods.

Stochastic methods

Stochastic methods use principles from other branches of science, such as biological systems, or physical chemistry. The main feature of these methods is that they can handle any degrees of nonlinearities and discontinuities.

Simulated annealing builds upon the behaviour of a physical system in a heat bath (Kirkpatrick et al., 1983). An ensemble of atoms can be found to different energy states. After reducing the temperature, the mobility of the atoms is lost, and the energy of the system decreases. When the system is frozen, the lowest energy state is taken. In a mathematical representation the variables can behave as the atoms in the physical system. The target is not the total energy, but another objective function. Simulated annealing is an iterative procedure. In each step of the algorithm a variable is given a small random perturbation, and the objective is calculated. If this is smaller than in the previous step, the perturbation is accepted. If the objective increased, then the perturbation is accepted with a calculated probability.

Evolutionary algorithms are based on the collective learning process within a biological population of individuals, each of which represents a search point in the space of potential solutions to a given problem (Bäck and Schwefel, 1993; Gross and Roosen, 1998). The population is arbitrarily initialized, and it evolves towards better and better regions of the search space by means of randomized processes of selection, mutation, and recombination. The environment (given aim of the search) delivers a quality information (fitness value) of the search points, and the selection process favours those individuals of higher fitness to reproduce more often than worse individuals. The recombination mechanism allows the mixing of parental information while passing it to their descendants, and mutation introduces innovation into the population. According to the above authors term 'evolutionary algorithm' covers three algorithms which differ only in the details: evolutionary programming, evolution strategies, and genetic algorithm.

Like simulated annealing, *tabu search* is an iterative neighbourhood search technique that attains the solution space by repeatedly performing state transitions from current state to a new state in its neighbourhood (Glover, 1989, 1990; Lin and Miller, 2004). The performed transitions are collected in a list, and the reverse moves of these transitions are associated with tabu status (i.e. forbidden) in order to force the search away from previous solutions and to prevent the search from getting trapped into a cycle. This method utilises selected ideas from artificial intelligence to decide on state transitions.

Deterministic methods

Deterministic methods are often called mathematical programming. These methods have three main steps (Grossmann, 1996). (1) First a so-called superstructure is generated, which contains all the considered alternative structures of the problem. (2) Then a rigorous

mathematical representation of the superstructure is formulated. (3) Finally the mathematical model is optimised.

There are two well-known mathematical programming methods: mixed integer nonlinear programming and generalized disjunctive programming.

Mixed integer nonlinear programming (MINLP) represents the superstructure using only algebraic equations. It contains continuous variables assigned to the operational parameters of a structure, and binary variables assigned to discrete decisions.

Generalized disjunctive programming (GDP) uses algebraic equations and also logical constraints (Raman and Grossmann, 1994). Operational parameters in this method are also denoted by continuous variables, but discrete decisions are expressed by logical variables.

Stochastic optimisation methods cannot guarantee finding the global optimum of a problem; however, they usually find a solution really near to the optimal one. Mathematical programming methods can guarantee global optimum in case of convex equations and search space, but in case of strong non-convexity they can be trapped in local optima.

1.1.4. Hybrid methods

The advantages of different approaches can be exploited by combining them.

Fonyó and Mizsey (1990; Mizsey and Fonyó, 1990) combined the hierarchical method of Douglas (1988) with mathematical programming. In the first step of their method, hierarchical level is used to create good preliminary flowsheets with simple energy integration. Then user-driven level is involved to tackle all type of constraints, complex configurations, and additional implicit knowledge derived during the hierarchical approach. Finally rigorous level is used to perform final design using mathematical programming.

Kravanja and Grossmann (1997) and Daichendt and Grossmann (1998) also integrated the mathematical programming approach using MINLP techniques and hierarchical decomposition heuristic approach. The main difference of their development from the results of Mizsey and Fonyó (1990ab) is that it is concerned with the conceptual design phase, and mathematical programming is used not only in the final design but it is integrated with hierarchical decomposition.

Mathematical programming approach was also integrated with the pinch analysis (Kravanja and Glavic, 1997), where composite curves were used in pre-screening of heat exchanger network superstructures.

Comeaux (2000) combined thermodynamic insights and mathematical programming in mass exchange network synthesis. Using stream data, and principle of vertical mass transfer, an insight based superstructure is generated which contains thermodynamically feasible matches only. Based on this superstructure, a pure nonlinear programming problem is formulated, and then optimised. Szitkai and co-workers (2005) further developed this technique, and published a new superstructure for mass exchange network synthesis based on the heat exchanger synthesis superstructure of Yee and Grossmann (1990).
Hostrup and co-workers (2001) also combined thermodynamic insights and mathematical programming approach using MINLP. They used the thermodynamic insight method of Jaksland et al. (1995) to generate the superstructure of the mathematical optimisation.
Fraga and Zilinskas (2003) combined local search methods for the continuous design parameters for the units of heat integrated distillation sequences; and evolutionary optimisation procedure for the design of the heat exchanger network.

In my dissertation I mainly deal with the mathematical programming approach. First, I study the use of case-based reasoning in process synthesis for the preparation of mathematical programming model formulation, namely in the generation of the superstructure. Then I study the relations between the superstructure, its graph representation, and the generated mathematical model. I present an automatic procedure to automatically generate an MINLP model based on the R-graph representation of the superstructure. This automatically derived model can serve as a reference in the comparison of MINLP models to decide whether an MINLP model represents the superstructure, or not. Then I present a method to enhance the characteristic of an MINLP model.
Before presenting my results, in the next sections, I give a detailed description about the used methods, case-based reasoning, and mathematical programming.

1.2. Case-based reasoning

As it was mentioned above, case-based reasoning imitates a human reasoning and tries to solve new problems reusing solutions that were applied to past similar problems. CBR deals with data from the previous situations, and reuses results and experience to fit a new problem situation.
The central notion of case-based reasoning is a case. The main role of a case is to describe a single event from past where a problem was solved. A case is made up of two components:

problem and solution. Typically, the problem description consists of a set of attributes and their values. Cases are collected in a set to build a case library (case base). The library of cases must roughly cover the set of problems that may arise in the considered domain of application.

The main phases of the case-based reasoning activities can be described typically as a cyclic process (see Fig. 1.1). During the first step, retrieval, a new problem (target case) is matched against problems of the previous cases (source cases) by calculating the similarity function, and the most similar problem together with its stored solution are found. If the proposed solution does not meet the necessary requirements of actual situation, then adaptation is the next step, and a new solution is created. The obtained solution might be validated by external rules or human. The approved solution and the new problem together build a new case that is incorporated in the case library during the learning step. In this way, CBR system evolves as the capability of the system is improved by extending the stored experience.

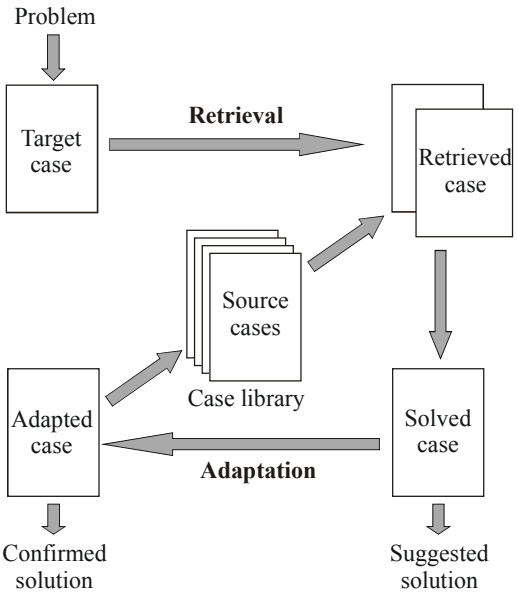


Figure 1.1. Case-based reasoning cyclic process

One of the most important parts of the CBR-cycle is the retrieval. During the retrieval, the attributes of the target cases are compared to find the most similar case. There are two widely used retrieval techniques (Watson, 1997): nearest neighbour and inductive retrieval. The nearest neighbour retrieval simply calculates the differences of the attributes, multiplied by a weighting factor. In inductive retrieval a decision tree is produced, which classifies the cases. There are classification questions about the main attributes in the nodes of the tree; by answering these questions the most similar case is found.

1.3. Mathematical programming

Mathematical programming has been well studied in the last decades. Several review papers have been published, such as Grossmann (1985, 1996), Floudas (1995), Grossmann and Kravanja (1995, 1997), Grossmann et al. (1999), Biegler and Grossmann (2004); and an overview of future perspectives (Grossmann and Biegler, 2004).

As it was mentioned above, mathematical programming has three main steps: (1) generation of the superstructure; (2) formulation of the mathematical representation; and (3) optimisation of the mathematical model. These steps are detailed in this section.

1.3.1. Generation of the superstructure

In the first step a superstructure has to be generated. This step is presented by an example. The example superstructure is shown in Fig. 1.2. In this problem the target is to produce final product *B* from raw material *A*. For this aim two reactors can be used, connected in parallel. Before the reactors the raw material has to be compressed to proper pressure, and heated to proper temperature. The end product of the reaction, which contains both components *A* and *B*, is separated using rectification. The top product of the rectification column is material *B*, which is taken as final product. The bottom product, material *A*, is recycled, and mixed to the raw material.

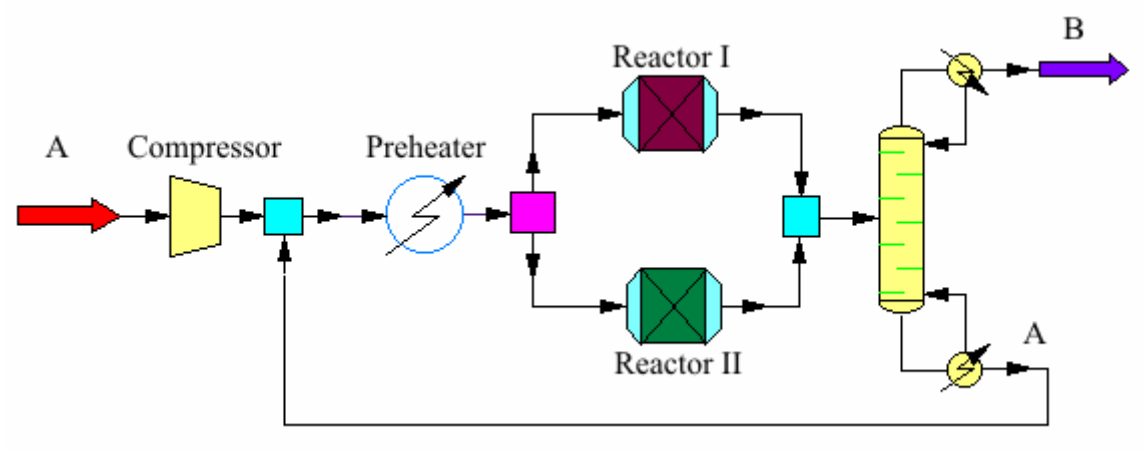


Figure 1.2. Example flowsheet

Mathematical representation usually is not written for the superstructure, but for the graph-like representation of the superstructure. The representations of superstructures have two main types (Yeomans and Grossmann, 1999a): state-task network (STN), and state-equipment network (SEN).

For the state-task network representation, first the states and tasks of the problem has to be defined (Table 1.1), and then the connections between them are represented in a graph (Fig. 1.3). Once the states and tasks are identified, it is necessary to determine what type of equipment can perform each task, and then to assign it to the corresponding task. There are two cases for this purpose:

- One task–one equipment (OTOE) assignment: in this case each task is assigned to a single equipment unit.
- Variable task–equipment (VTE) assignment: in this case, a set of equipment that can perform all the tasks needed in the flowsheet is identified first. The assignment of the equipment to the tasks is then considered as part of the optimisation model.

The STN representation in Fig. 1.3 is an OTOE assignment.

Table 1.1. States and tasks in the process

states		tasks	
1	raw material *A* at low pressure and low temperature	1	compression of raw material *A*
2	raw material *A* at low pressure and low temperature	2	mixing raw material *A* and recycled material *A*
3	mixture *A* at high pressure and low temperature	3	heating mixture *A*
4	mixture *A* at high pressure and high temperature	4	splitting mixture *A*
5	splitted mixture *A* into Rector I.	5	reaction of mixture *A* in Reactor I.
6	splitted mixture *A* into Rector II.	6	reaction of mixture *A* in Reactor II.
7	product from Reactor I.	7	mixing the products of reactions
8	product from Reactor II.	8	separation of mixed products of reaction
9	mixed product		
10	pure material *B*		
11	recycled material *A*		

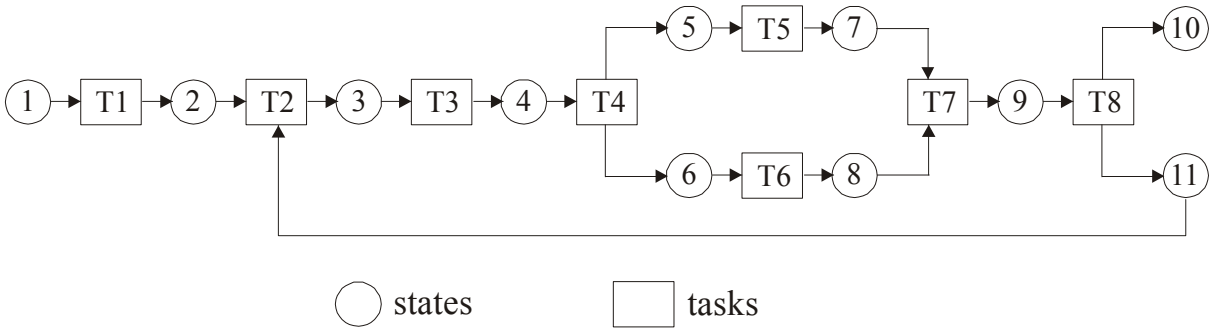


Figure 1.3. STN representation of the structure

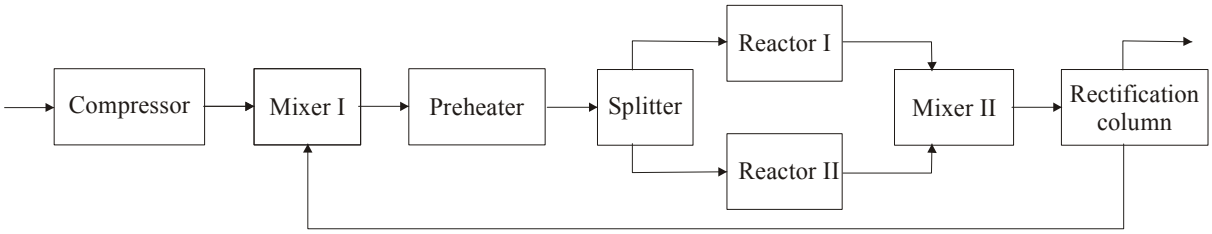


Figure 1.4. SEN representation of the structure

In the state-equipment network first the necessary units of the process are defined. Then these units are connected in a graph-like representation (Fig. 1.4). The units are the nodes of the graphs, and the directed edges are the streams between them. The tasks that can take place in a specific equipment are not pre-specified, which is equivalent to a VTE assignment.

The superstructure has to be generated in a way that it contains all the considered structures. It is usually based on engineering experience.

Friedler and co-workers (1993) suggested to generate the superstructure in a combinatorial way. They defined P-graph, an STN representation, for representing structures (Friedler et al., 1992a). P-graphs form a special class of bipartite graphs; they consist of operational unit nodes (O-type nodes) and material nodes (M-type nodes), connected by edges. Edges always connect two different kinds of nodes, namely unit nodes and material nodes. According to the P-graph approach, material nodes represent some predefined composition domains. A domain may be assigned by dominance, or practical lack of some components as a special, perhaps informal, assignment of the composition domain. Generally, some property domain is predefined. Operational units are imagined as entities transforming a set of material states in the domains represented by material nodes into another set of material states in domains also represented by the corresponding material nodes.

A P-graph represents a combinatorial possible structure if it satisfies the following axioms:

1. Every final product is represented in the graph.
2. A node of M-type has no input if and only if it represents a raw material.
3. Every node of O-type represents an operating unit defined in the synthesis problem.
4. Every node of O-type has at least one path leading to a node of M-type representing a final product.
5. If a node of M-type belongs to the graph, it must be an input to or output from at least one node of O-type in the graph.

Using P-graphs, the superstructure (or maximal structure) automatically can be generated by the MSG algorithm (algorithm for Maximal Structure Generation), and the set of feasible structures by the SSG algorithm (Generation of the Solution-Structures) (Friedler et al., 1992b).

Friedler and co-workers (1998) demonstrated the use of P-graph in process network synthesis, and the generation of the MINLP model. Brendel and co-workers (2000) showed that the generation of the conjunctive and disjunctive normal forms to solve process synthesis problems by a logical formulation can be mathematically established on the basis of the combinatorial approach.

Our workgroup defined superstructure with mathematical rigor for general use in process synthesis (Rév et al., 2005). For this aim, we invented a special kind of graph, the so-called R-graph. The R-graph is a one task–one equipment (OTOE) graph. The nodes of an R-graph are not units but the input and output ports of the possible units. The directed edges correspond to streams; they always start from an output port node of a unit, and end on an input port node of a unit. The output port nodes are treated as arbitrary stream splitters, whereas the input nodes as arbitrary unifiers. The R-graph representation of the example flowsheet is shown in Fig. 1.5.

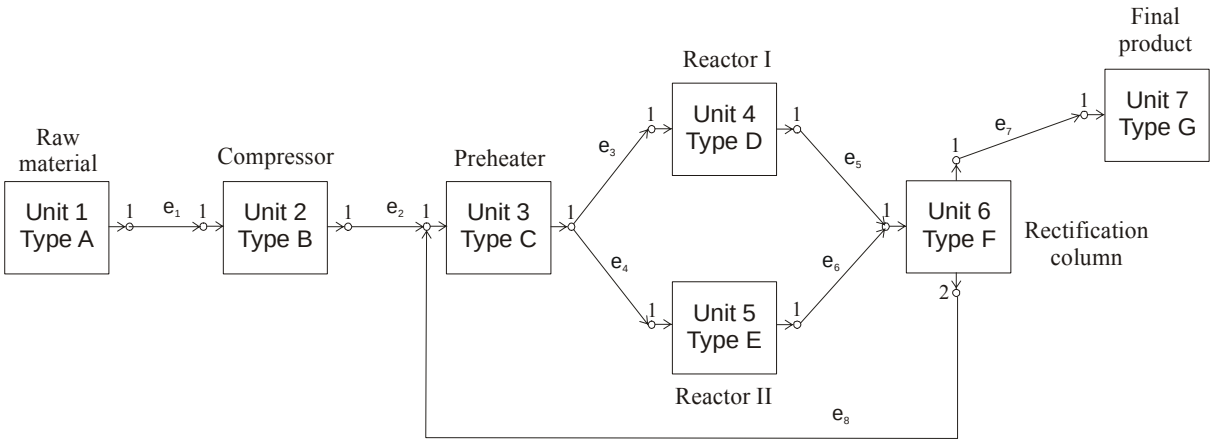


Figure 1.5. R-graph representation of the example flowsheet

An R-graph is a graph also in mathematical sense. All the edges start from a node, and end on a node. Therefore, source and sink units are included in order to prevent edges to start from or end on outside the graph. The source units (e.g. Unit 1 in Fig. 1.5) have no input nodes; the sink units (e.g. Unit 7 in Fig. 1.5) have no output nodes.

The subgraph of an R-graph is a short-hand for sub-R-graph of an R-graph, i.e. it is also an R-graph. All the nodes have to be connected in a subgraph, i.e. in an sub-R-graph. A counter-example is shown in Fig. 1.6, where the second output node of Unit 6 is not connected to any other node. This is not an R-graph even if the engineer can easily assign meaning to this figure by considering the stream of the unoccupied port as a product, which is not recycled. If that kind of layout is permissible then a source unit type should also be used in the graph connected to that certain output port.

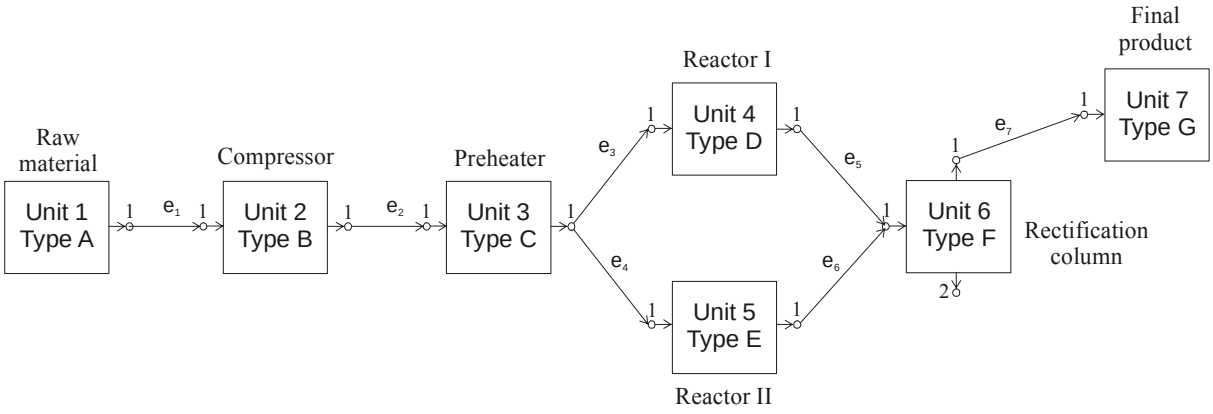


Figure 1.6. A counter-example. This is not an R-graph because there is an unconnected node.

Structural multiplicity is an important phenomenon caused by the possibility of representing the same structure with different graphs. For example, if Reactor I and Reactor II had the same type (Type DE) in our example, then the two subgraphs in Fig. 1.7 would represent the same structure. But they are different because the nodes and edges of the supergraphs are unambiguously denominated. (Graphs, by mathematical sense, are constructed from labelled entities.) These two R-graphs are called isomorphic because they are identical, neglecting the differences in different copies of units of the same type. In other words, they are isomorphic because one of them can be transformed to the other one merely by re-naming the units. Consequently, the structures are not equivalent to graphs but to sets of isomorphic (or, in other word: equivalent) graphs.

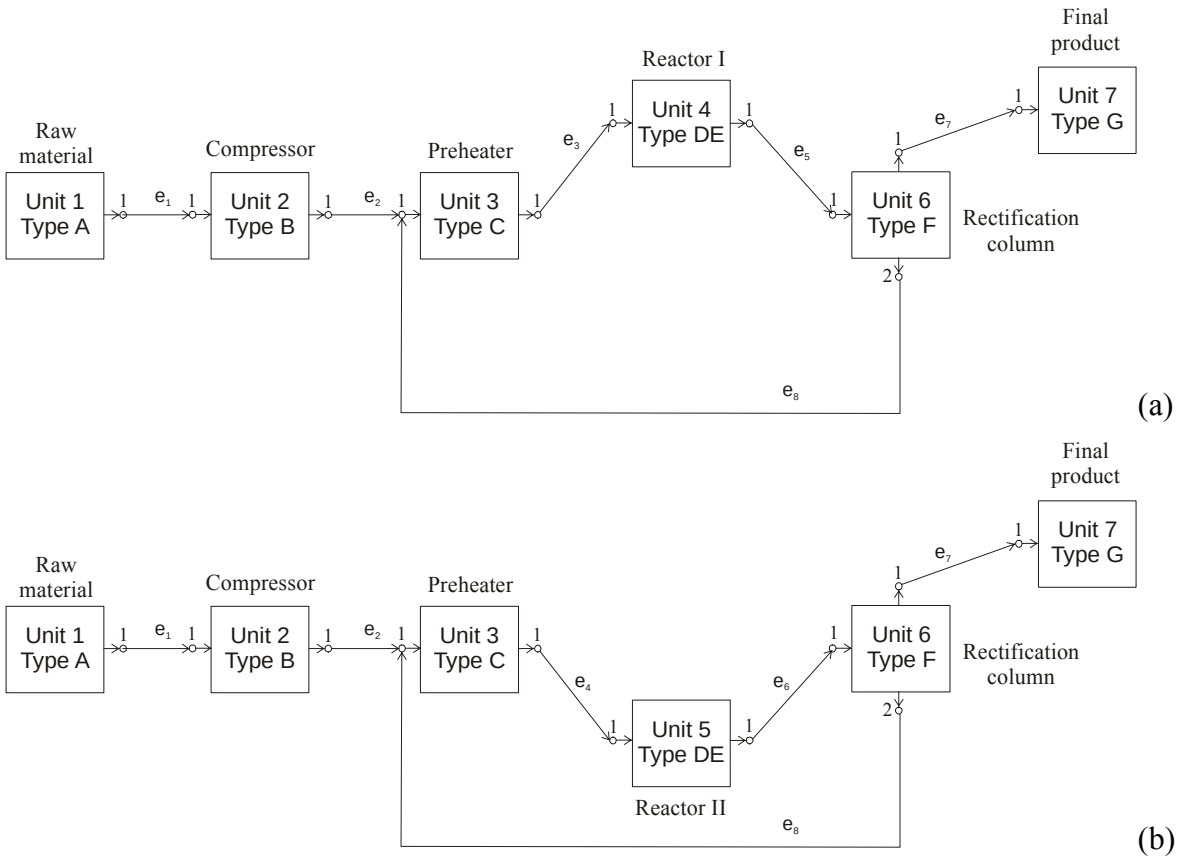


Figure 1.7. Isomorphic R-graphs

Based on this definition, a structure s is the superstructure of structures s_1, s_2, …, if these structures are the substructures of s. The R-graph representations of the substructures are the sub-R-graphs of the R-graph representation of the superstructure.

A benefit of the R-graph representation is that it is close to the modular unit approach, where not the units but the their input and output ports are connected. The main difference is the lack of stream splitters and unifiers, but it is beneficial for avoiding by-pass redundancy during optimisation.

Fig. 1.8 serves as a simple (and arbitrary) example for demonstrating the redundancy related to by-passes. (Similar figure can be seen e.g. in Renaume et al., 1995) By-passing unit A is necessary to let unit B exist even if A is not included in the structure. By-passing unit B is necessary for the reverse reasoning. An identical substructure at the lower branch may occur. For our didactical purposes we apply just a unit C there, also bypassed. These units and streams form the superstructure. Exclusion of a unit from the final structure does not exclude its by-pass stream. Thus, excluding unit C may bring to life the structure shown in Fig. 1.9.

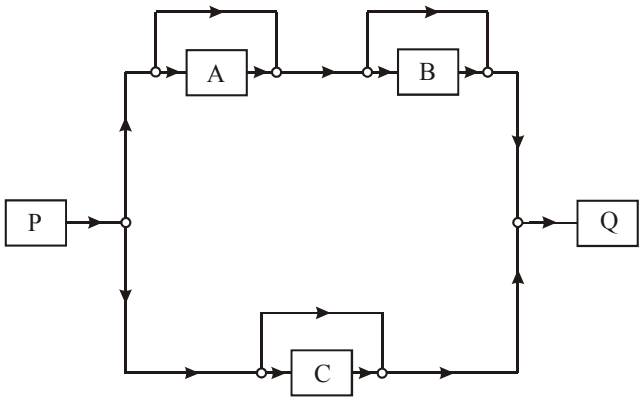


Figure 1.8. Branched structure with by-passes

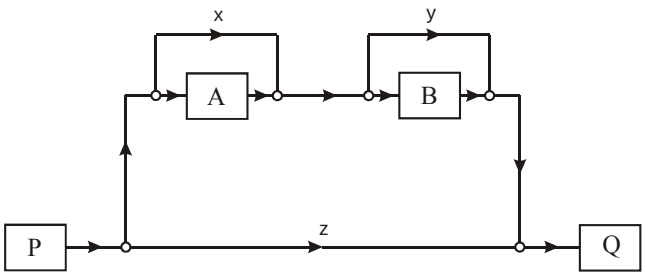


Figure 1.9. A structure with redundant by-passes

Let the flow rates of the streams leaving units P, A, and B are given; and let the flow rates of streams entering into units A, B, and Q are also given. For the sake of simplicity and clarity, suppose that the flow rates x and y are equal: $x=y$. Then the sum $x+z$, equalling the sum $y+z$, is a constant, where z is the flow rate of the stream by-passing the upper branch. For example, let the flow rate leaving Unit P be 100, $z=50$, $x=20$, and the input to Unit A be 30. In the same time $y=20$, and let the output from Unit B be 35, then the input to Unit Q is 105.

Then the flow rate z can be changed on the cost of simultaneously changing the values of x and y, without having any influence on the units. Without changing the input and output of units P, A, B, and Q, the flow rates may be, for example, $z=60$, $x=y=10$, or $z=30$, $x=y=40$.

As a result, the objective function (not given here) may remain constant in a continuous subdomain of the feasible solutions. This phenomenon, that has a detrimental effect on optimisers, is called *by-pass redundancy*.

1.3.2. Mathematical representation of the superstructure

After the superstructure is generated, its mathematical representation has to be formulated. As it was mentioned above, two kinds of mathematical representations are in use: mixed integer nonlinear programming (MINLP), and generalized disjunctive programming (GDP).

An MINLP problem contains both continuous and integer variables, and the integer variables are usually binary variables. An MINLP problem can be formulated as follows (Kocis and Grossmann, 1987; Grossmann, 1996):

$$\begin{aligned} \min \quad & OBJ = \mathrm{f}(\mathbf{x},\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x},\mathbf{z}) \le \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} = \{\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \mathbf{L} \le \mathbf{x} \le \mathbf{U}\} \\ & \mathbf{z} \in \mathbf{Z} = \{\mathbf{z} \mid \mathbf{z} \in \{0,1\}^k,\ \mathbf{A}\mathbf{z} \le \mathbf{a}\} \end{aligned} \tag{1.1}$$

where $\mathbf{x}$ is the vector of continuous variables specified in the range $\mathbf{X}$, and $\mathbf{z}$ is the vector of binary variables which must satisfy linear integer constraints $\mathbf{Az} \le \mathbf{a}$. $\mathrm{f}(\mathbf{x},\mathbf{z})$, and $\mathbf{g}(\mathbf{x},\mathbf{z})$ are real functions; they may be nonlinear. The commonly applied solver algorithms usually enable the binary variables appearing in linear members only.

In the conventional representation, binary variables (z_m) denote the existence of units (m=1,…M). If the unit m exists in the actual structure, then z_m=1, otherwise z_m=0. The logical relations between units can be expressed in algebraic form by using binary variables (Raman and Grossmann, 1991, 1993).

In GDP, continuous variables are used for the representation of operational parameters like pressure, temperature, etc. The discrete decisions (such as describing whether a unit exists or not) are formulated by logical expressions using logical variables (Raman and Grossmann, 1994; Grossmann and Daichendt, 1996; Grossmann and Türkay, 1996; Yeomans and Grossmann, 1999b). The general form of a GDP problem is as follows:

$$\begin{aligned} \min \quad & OBJ = \sum_{m \in \mathbf{M}} c_m + \mathrm{f}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x}) \le \mathbf{0} \\ & \begin{bmatrix} Z_m \\ \mathrm{h}_m(\mathbf{x}, c_m) \le 0 \end{bmatrix} \vee \begin{bmatrix} \neg Z_m \\ \mathbf{x} = 0 \\ c_m = 0 \end{bmatrix} \qquad m=1,\ldots M \\ & \mathbf{x} \in \mathbf{X} = \{\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \mathbf{L} \le \mathbf{x} \le \mathbf{U}\} \\ & \mathbf{Z} \in \overline{\mathbf{Z}} = \{\mathbf{Z} \mid \mathbf{Z} \in \{\text{true, false}\}^k,\ \Omega(\mathbf{Z}) = \text{true}\} \end{aligned} \tag{1.2}$$

where x and c_m are continuous variables, the latter being used to model costs associated with the units; $\mathbf{g}(\mathbf{x})$ represents constraints that are valid over the entire search space; Z_m is logical variable denoting the existence of unit m.

If the logical variable Z_m is true, then unit m exists in the actual structure, and the operational equations of the unit ($h_m(\mathbf{x},c_m)\leq 0$) are satisfied. If Z_m is false ($\neg Z_m$), then the unit does not exist; in that case the variables, and the cost of the unit, take zero value. These two sets of equations are connected with a logical 'or' ($\vee$) relation.

The relations between units can also be described with pure logical relations ($\Omega(\mathbf{Z})$=true) using logical operators like 'and' ($\wedge$), 'or' ($\vee$), 'exclusive or' ($\oplus$), and 'implication' ($\Rightarrow$) (see Raman and Grossmann, 1991; Hooker et al., 1994). For example, in our example two reactors are connected in parallel. According to engineering experience, the use of both reactors must be uneconomic. Therefore, a logical relation is necessary to express that they cannot exist simultaneously. This can be done using the logical operator 'or':

$$Z_I \vee Z_{II} \tag{1.3}$$

A GDP representation can be transformed into MINLP representation by using binary variables (z_m) instead of logical ones (Z_m), and transforming the logical relations into algebraic form. There are three well-known techniques for this transformation (Balas, 1985; Hui, 1999): Big M method, Multi M method, and Convex hull method. These methods are represented by a simple example problem transforming a disjunctive expression (Eq. 1.4) into algebraic equations:

$$\begin{bmatrix} Z_m \\ c_m = d_m \\ L \leq x \leq U \end{bmatrix} \vee \begin{bmatrix} \neg Z_m \\ c_m = 0 \\ x = 0 \end{bmatrix} \tag{1.4}$$

where d_m is a fixed constant; L and U are the lower and upper bounds on x, respectively.

Eq. 1.4 can be transformed into algebraic equations using Big M method in the following way:

$$-M(1-z_m) \leq c_m - d_m \leq M(1-z_m) \tag{1.5a}$$

$$c_m \leq M \cdot z_m \tag{1.5b}$$

$$-M \cdot z_m \leq x \leq M \cdot z_m \tag{1.5c}$$

where M is a Big M value. If unit m exist, i.e. z_m=1, then both sides in Eq. 1.5a are equal to 0; therefore, c_m takes the value d_m. In this case variable c_m has to be smaller than M (Eq. 1.5b), and the value of x has to be in the interval [-M; M] according to Eq. 1.5c.

If unit m does not exist, i.e. z_m=0, then c_m does not have to take value d_m, the difference between them can take any value from interval [-M; M] according to Eq. 1.5a. In this case variable c_m has to take value 0 (Eq. 1.5b), and so does variable x (Eq. 1.5c).

The Big M value (M) should be the greatest value of the denoted lower and upper bounds, i.e. all the above expressions and variables in Eqs. 1.5 should be able to take value above their lower and below their upper bounds. The main drawback of this method is that in all the equations the same Big M value is used; therefore, the relaxation of the model is poor.

The Multi M method has just one main difference from the Big M method; it uses different Big M values for each expressions and variables:

$$L_m^{c-d}\left(1-z_m\right) \le c_m - d_m \le U_m^{c-d}\left(1-z_m\right)$$

$$c_m \le U_m^{c} \cdot z_m$$

$$L_m^{x} \cdot z_m \le x \le U_m^{x} \cdot z_m \tag{1.6}$$

where L and U are the lower and upper bounds, respectively, given to the actual expression or variable.

In this way, the relaxation of the model is greatly improved. Note that in the literature, when the Multi M method is used then usually it is also called Big M method, and the original Big M method is not in use.

In Convex hull method, all the variables are disaggregated into continuous variables assigned to each term of the disjunction, and the constraints in the disjunction are written for these disaggregated variables:

$$c_m = c_m^1 + c_m^2$$

$$x = x^1 + x^2$$

$$c_m^1 = d_m \cdot z_m$$

$$c_m^2 = 0 \cdot \left(1-z_m\right)$$

$$L \cdot z_m \le x^1 \le U \cdot z_m$$

$$x^2 = 0 \cdot \left(1-z_m\right) \tag{1.7}$$

If the unit m exist, i.e. z_m=1, then the disaggregated variables with superscript 1 (c_m^1 and x^1) satisfy the equations of the first disjunction in Eq. 1.4, and the other disaggregated variables take zero value. If the unit does not exist, i.e. z_m=0, then the disaggregated variables with superscript 2 satisfy the equations of the second disjunction in Eq. 1.4, and the disaggregated variables with superscript 1 take zero value.

The pure logical constraint in GDP representation, $\Omega(\mathbf{Z})$=true, can also be transformed into algebraic equations using binary variables (Raman and Grossmann, 1991). For example, the logical constraint between the logical variables of the reactors in our example (Eq. 1.3) can be transformed into algebraic form in the following way:

$$z_I + z_{II} \leq 1 \tag{1.8}$$

According to Eq. 1.8, both the binary variables value cannot take in the same time 1, i.e. the reactors cannot exist simultaneously.

1.3.3. Optimisation algorithms

In the final step of mathematical programming, the formulated mathematical problem has to be optimised. Some general overviews about the optimisation algorithms are: Biegler et al. (1997), Floudas (1995), Grossmann and Kravanja (1997), Grossmann (1996), Grossmann et al. (1999).

There are five main MINLP algorithms (Grossmann, 1996): branch and bound, outer approximation, generalized Benders decomposition, extended cutting plane, and LP/NLP based branch and bound.

The *branch and bound algorithm* (Borchers and Mitchell, 1994) starts by solving the continuous NLP relaxation of the original MINLP problem. That is, continuous variables with lower bound 0, and upper bound 1 are used instead of the binary ones in the relaxed NLP problem. If all the binary variables take binary values, the search is stopped. Otherwise, it performs a tree search in the space of the binary variables. These are successively fixed at the corresponding nodes of the tree, giving rise to relaxed NLP subproblems which yield lower bounds for the subproblems in the descendant nodes. Fathoming of nodes occurs when the lower bound exceeds the current upper bound, or when all integer variables take binary values. The latter yields an upper bound to the original problem. This method is attractive only if the NLP subproblems are relatively inexpensive to solve, or only a few of them need to be solved.

In the *outer approximation method* (Duran and Grossmann, 1986), NLP and MILP subproblems are solved iteratively. The NLP subproblem is derived from the original MINLP problem by fixing the binary variables. The MILP subproblem is derived from the MINLP problem by linear relaxation of equations using the solutions of the previous NLP subproblems. This relaxation approximates the functions from below, and the solution space from outside. In each iteration an integer cut is added to the MILP subproblem in order to

exclude the solutions of previous iterations, i.e. to prevent the search from getting trapped into a cycle. The NLP subproblems yield an upper bound; the MILP subproblems yield a lower bound. The cycle of iterations is stopped if the lower and upper bound are within a tolerance, or if the MILP subproblem becomes infeasible. This method generally requires relatively few cycles (major iterations).

The *generalized Benders decomposition method* (Geoffrion, 1972) is similar to the outer approximation method. The only difference is in the way the MILP subproblem is defined. In this method only active inequalities are considered, and the set of continuous variables is disregarded. This method usually needs more iterations than the previous method to find the optimum.

In the *extended cutting plane method* (Westerlund and Petersson, 1995), only MILP subproblems are solved iteratively. In each iteration the most violated constraint at the predicted point is added. The cycle of iterations stops if the maximum constraint violation lies within a specified tolerance. Since the continuous and binary variables are converged simultaneously, a large number of iterations may be required.

The *LP/NLP based branch and bound method* (Quesada and Grossmann, 1992) avoids the complete solution of the MILP subproblems in each major iteration. An LP-based branch and bound method is performed for the MILP subproblems, solving relaxed NLP subproblems at those nodes in which feasible integer solutions are found.

GDP problems can be solved by transforming them into MINLP problems, and then an MINLP algorithm can be used. However, this method does not exploit the disjunctive structure of the model.

Türkay and Grossmann (1996) proposed an algorithm for solving nonlinear GDP models involving two terms in each disjunction. This is a logical-based outer approximation algorithm. Its main advantage is that the NLP subproblems are generated in a way that only the active constraints are considered. If in an NLP subproblem Z_m is false, i.e. unit *m* does not exist then those constraints which have to be satisfied in the case when unit *m* exists, are not considered. The MILP subproblems are obtained by convex hull linearization of the nonlinear inequalities.

Using the algorithm of Lee and Grossmann (2000), GDP problems involving more than two terms can also be solved. In this method, first the convex relaxation of the original GDP problem is generated, based on the convex hull of each nonlinear disjunction. Then a special branch and bound algorithm is used. In the branching, that disjunction term is selected which

is closest to the optimum of the convex relaxation problem. The convex hull relaxation of the remaining disjunctions, which have not examined yet, is at the other branch.

In my dissertation, I use the MINLP formulation of the problems. The MINLP problems are solved with GAMS program (Brooke et al., 1992) using DICOPT++ solver (Viswanathan and Grossmann, 1990). This solver performs the above mentioned outer approximation algorithm.

1.4. Examples for the use of MINLP

In this section, some process synthesis examples using MINLP are shown. The examples are taken from the literature, and represent the maybe most often used processes: heat exchanger networks, mass exchange networks, and distillation.

1.4.1. Heat exchanger networks

The basic HEN synthesis problem can be formulated as follows (Biegler et al., 1997):

Given

- a set of hot process streams to be cooled and a set of cold process streams to be heated;
- the flowrates and the inlet and outlet temperatures for all these process streams;
- the heat capacities for each of the streams versus their temperatures as they pass through the heat exchange process;
- the available utilities, their temperatures, and their costs per unit of heat provided or removed.

The goal is to determine the heat exchanger network for energy recovery that will minimize the annualized cost of the equipment plus the annual cost of utilities.

For the synthesis of heat exchanger synthesis problem, the most known approach is perhaps the pinch analysis (Linnhoff, 1993, 1994). As it was mentioned above, it is a targeting method. First the minimum utility assumptions, then the connections of the hot and cold streams, the number and the size of heat exchanger units, are determined.

The main disadvantage of this approach is that it does not guarantee the reach of global optimum because using the minimum consumption usually does not result minimum total cost. If more is used from the hot utility, for example, than minimum, then, providing the same

amount of heat transfer, the temperature changing of this hot utility will be smaller. Therefore, the average temperature difference in the heat exchanger will be greater, and a smaller unit will be enough. Using more hot utility the operating cost increases, but because of the smaller heat exchanger, the equipment cost decreases. These two cost effects can result in a decrease of the total cost.

This problem can be solved by using optimisation techniques, because in this case the synthesis and the minimization of the total cost are performed simultaneously. For this aim, Yee and Grossmann (1990) developed a proper superstructure and MINLP model. The superstructure, consisting two hot streams (H1 and H2) and two cold streams (C1 and C2) are shown in Fig. 1.10. Streams are driven in a counter-current way. In this case, two temperature interval (k=1 and 2) exist in the superstructure. Streams are splitted in each temperature interval. In this way, each cold stream can be matched to each hot stream in each interval. These possible matchings are denoted with circles. At the end of the intervals the branches of the splitted streams are unified, and are driven to the next temperature interval.

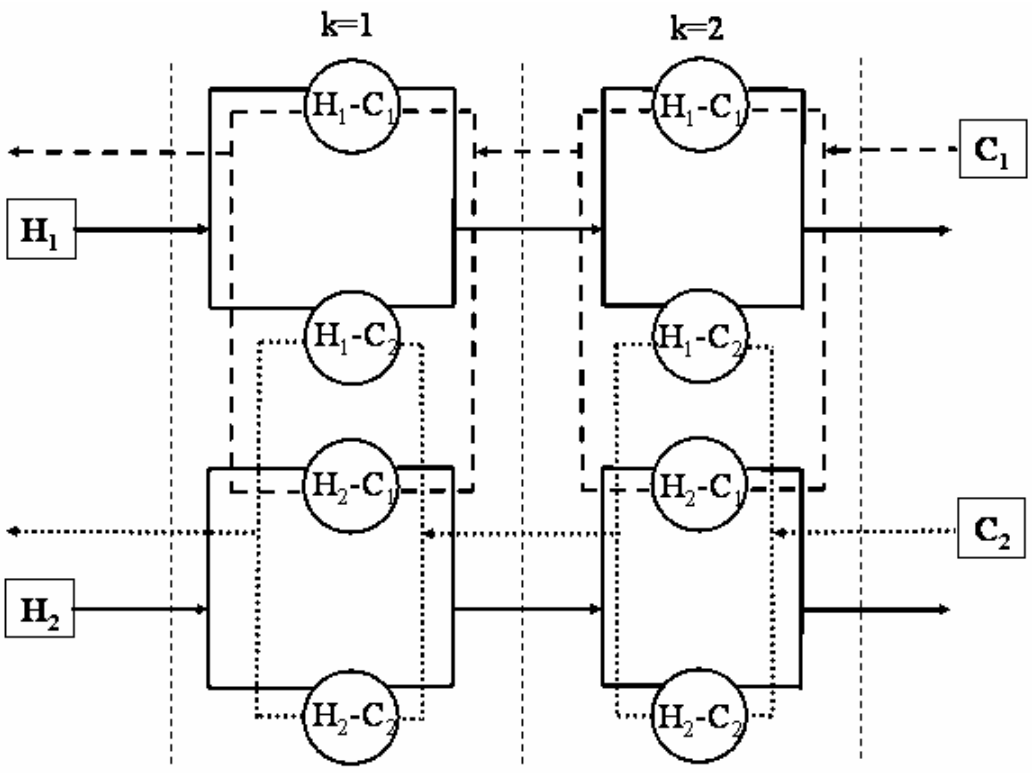


Figure 1.10. Superstructure of Yee and Grossmann

In the MINLP model, a binary variable represents the existence of a heat exchanger unit in each connection. Temperature, heat power and the flowrate of the external streams, are

represented with continuous variables. The model involves the heat balance of each heat exchanger and each heat interval, the definition of the known temperatures, the calculation of the transfer between the hot and cold process streams, the logical constraints, and the calculation of the temperature differences at the connections. Only the streams with the same temperature are allowed to be mixed; therefore, almost all the equations are linear. Only the calculation of the average temperature difference is, and cost function may also be nonlinear.
Using the MINLP model of Yee and Grossmann, the synthesis of heat exchange networks can be performed in one step. Since almost all the equations are linear, finding the global optimum has a great chance.

1.4.2. Mass exchanger networks

Mass exchange networks (MENs) are systems of interconnected direct-contact mass-transfer units that use process lean streams or external mass separating agents (MSAs) to selectively remove certain components (often pollutants) from rich process streams. In context of the overall process, the MEN is usually a part of the separation network. The main function of the MENs is to fully exploit the on-site cleaning capacities of the chemical facilities, hence MENs achieve environmental protection goals through process integration.
The first, pinch-based, solution methodology of El-Halwagi and Manousiouthakis (1989) was extended by Hallale and Fraser (2000ab). Using the advanced targeting methods of Hallale and Fraser, both the capital cost and the total annual cost (TAC) of the network can be predicted ahead of any design. Still, pinch technology for mass exchange network synthesis (MENS) does not provide with a systematic way for deriving the optimal network structure. The network design includes trial and error elements, especially when large or multiple component problems are considered.
Sequential mathematical programming approaches, that are mainly automated versions of a pinch design technique, were developed to facilitate the targeting step in case of large problems where the graphical approach of the pinch method is not convenient to use (El-Halwagi, 1997; Grossmann et al., 1999). The attribute “sequential” denotes that the synthesis is still decomposed into targeting and design steps. As a consequence, the trade-off between investment and operating costs is not taken into account rigorously.
The first mathematical programming method for mass exchange network synthesis was published by Papalexandri and co-workers (1994). The superstructure of their model has the following characteristics:

- The connection of a rich and a lean stream can correspond to a mass exchanger unit.
- Every inlet streams are splitted towards the possible mass exchanger units.
- Two streams can match more than once. In this way the system has more subsystems. For example, a pinch point divides the whole concentration interval into to two subintervals. But mass transfer should not be performed between these subintervals, according to the pinch technology. Therefore, before and after the pinch point, streams can have more than one connections.
- Before each possible mass exchanger units a mixer exists in which the streams from the inlet streams and the by-pass streams are mixed.
- After each possible connection, a splitter exists which split the outlet stream of the mass exchanger unit to a stream towards the final mixer, and to by-pass streams towards other mass exchanger units.

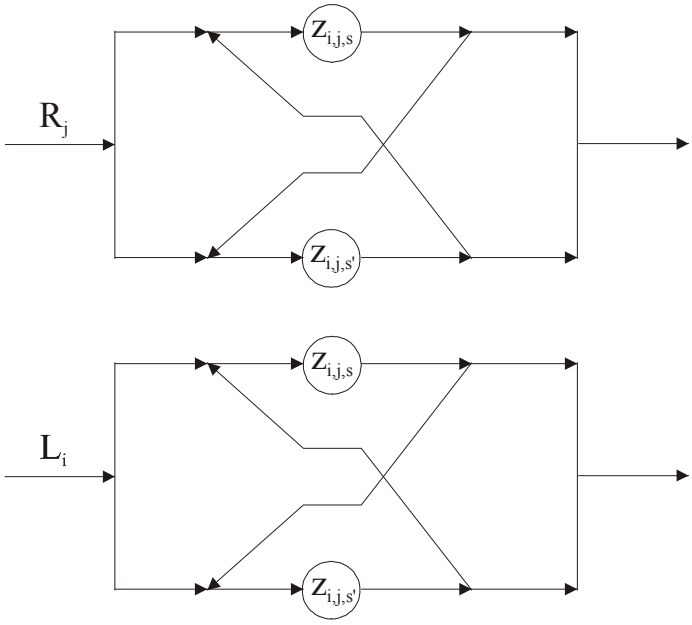


Figure 1.11. Superstructure of Papalexandri and co-workers

In Fig. 1.11 a part of the superstructure is shown, which represent the possible connections between rich stream j (R_j) and lean stream i (L_i). There are two subsystems in the superstructure denoted by s and s'. The existence of the possible mass exchanger units are denoted by binary variables $y_{i,j,s}$.

An MINLP model has been developed by Szitkai and co-workers (2005), based on the analogy between HENs and MENs using the superstructure and model of Yee and Grossmann

(1990). Both kinds of networks transport a certain quantity among sources and sinks due to the existence of driving forces, as it is shown in Table 1.2.

Table 1.2. Analogy between MENS and HENS

	MENS	HENS
Transported quantity	Mass	Heat
Driving force	Concentration difference	Temperature difference
Source	Rich process streams	Hot process streams
Sink	Lean process streams	Cold process streams

Although the existence of an analogy between HENS and MENS is apparent, there are a few fundamental differences between the two synthesis problems that have to be considered when constructing a mathematical programming model for the synthesis of MENs. These differences are the following:

1. In case of MENS, equilibrium relations between the rich and the lean streams have to be taken into account when calculating the driving forces of the mass transfer, or when sizing the mass exchangers.
2. Analogy between HENS and MENS can be drawn only in case of single component MENS problems. The heat analogue of multiple components does not exist.
3. The problem statement of MENS (El-Halwagi, 1997) includes the determination of the process lean stream flow rates, while in case of the HENS problem, the cold stream flow rates are given a priori.
4. In case of MENS problems, it makes no sense defining external rich streams.

Resulting from point 1, phase equilibrium calculation is to be added to the mathematical programming model of Yee and Grossmann (1990). Point 2 requires the addition of non-linear component mass balances or necessitates the application of a modelling technique that can account for multiple components. Because of point number 3, the lean stream balances (flow rates multiplied by concentrations) become non-linear. The analogous constraints, representing cold stream balances in the HENS model of Yee and Grossmann (1990) are linear. Allowing for point number 4 results in a slightly different superstructure compared to that of Yee and Grossmann (Fig. 1.10).

1.4.3. Distillation columns

Distillation is one of the most widespread processes applied for separating multicomponent liquid mixtures. It is used for working up great volume; and it requires high investment and operation outlays. Therefore, the significance of the design of economically optimal distillation processes is of no question.

Optimising single columns is a well-known procedure; all the basic textbooks outline how to do it in an easy manner (Coulson et al., 1985). After approximately determining the minimum and the estimated optimal number of theoretical stages, optimising over the continuous variables is performed at varied values of the discrete variables. This is a 2-dimension discrete array of continuous optimisation tasks in the case of a single-feed, two-product column, because there are merely two column sections in this case. This task becomes much more difficult if several feeds and side-products are to be taken into account. The real problem, however, is synthesizing a distillation sequence, or a system of advanced distillation columns, or a complex flowsheet containing distillation units, when the number of distillation columns and their connections are not known in advance.

Recent reviews of the topic are Barttfeld et al. (2003), and Grossmann et al. (2005).

Mathematical formulations that represent rigorous distillation column configuration, fall into two categories: (1) one task–one equipment (OTOE) representations and (2) variable task–equipment (VTE) representations (Yeomans and Grossmann, 1999b).

In the OTOE representation, each stage has one task to work as an equilibrium stage. If the actual stage is not performed in a structure, than the stage is by-passed (Viswanathan and Grossmann, 1993ab). Such a superstructure is shown in Fig. 1.12. A boiler is located below the lowest stage (1st stage), and a condenser is located above the top stage of the column (J^{th} stage). Binary variables (z_j^{ref} and z_j^{bu}) represent the location of the returning streams after condensation and boiling, i.e. the stages where the reflux stream (*ref*) and the reboiled vapor stream (*bu*) are led to, beside the feed location. If z_j^{ref}=1, then the reflux stream is led to the j^{th} stage. Similarly, if z_j^{bu}=1, then the reboiled vapor stream is led to the j^{th} stage. Stages above the reflux stream location and below the reboiled vapor stream location are considered non-existing stages; whereas all the other stages exist. The MINLP model involves the total and component mass balances, the enthalpy balances, and the equations of the physical-chemical equilibrium, at each stage.

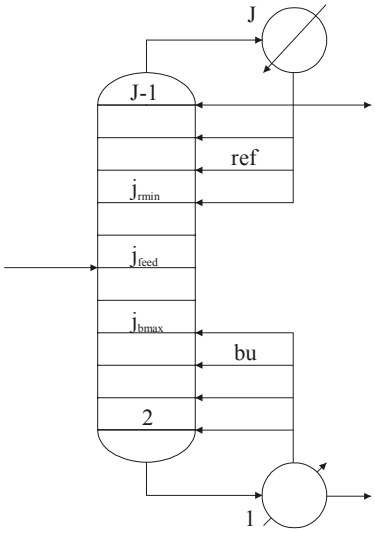


Figure 1.12. Superstructure of Viswanathan and Grossmann

In the OTOE approach, there are three main different arrangements (Barttfeld et al., 2003): (1) The feed location is fixed, and the stages where the boil-up and reflux flows enter the column are variables. (2) The position of the feed and the reflux inlet are variable, whereas the boil-up location is fixed. (3) The location of the feed and the boil-up streams are variables, and the inlet point of the reflux is fixed. The first arrangement is shown in Fig. 1.12.

In the VTE representation, each stage has two tasks: (1) an equilibrium stage, or (2) an input-output operation in the stage with no mass transfer. Logical or binary variables assign which task of a stage is performed in a structure (Yeomans and Grossmann, 2000ab).

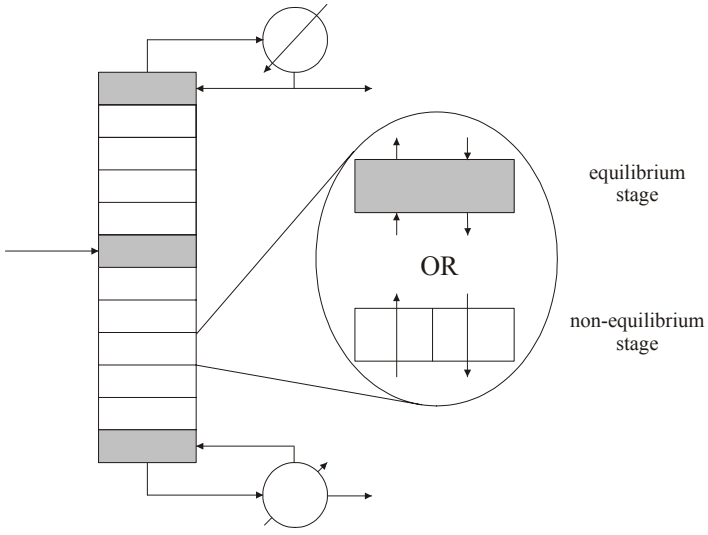


Figure 1.13. Superstructure of Yeomans and Grossmann

$$\begin{bmatrix} Z_j \\ \wedge \\ f_{c,j}^{L} = \mathrm{f}\left(T_j^{L}, P_j, \mathbf{x}\right) \\ f_{c,j}^{V} = \mathrm{f}\left(T_j^{V}, P_j, \mathbf{y}\right) \\ f_{c,j}^{L} = f_{c,j}^{V} \\ L_{c,j} = LIQ_j \cdot x_{c,j} \\ V_{c,j} = VAP_j \cdot y_{c,j} \\ T_j^{L} = T_j^{V} \end{bmatrix} \vee \begin{bmatrix} \neg Z_j \\ \wedge \\ f_{c,j}^{L} = 0 \\ f_{c,j}^{V} = 0 \\ x_{c,j} = x_{c,j+1} \\ y_{c,j} = y_{c,j-1} \\ L_{c,j} = L_{c,j+1} \\ V_{c,j} = V_{c,j-1} \\ T_j^{L} = T_{j+1}^{L} \\ T_j^{V} = T_{j-1}^{V} \end{bmatrix} \qquad \begin{matrix} c = 1,\ldots,\mathsf{C} \\ j = 1,\ldots,\mathsf{J} \end{matrix} \tag{1.9}$$

In the superstructure of Yeomans and Grossmann (2000a) the location of the feed stage, of the condenser, and of the reboiler are fixed (Fig. 1.13). In the GDP representation if the logical variable Z_j is true then the j^{th} stage works as an equilibrium stage. In this case the liquid and vapor fugacities are calculated, they are equal, and the concentration of the outlet streams depend on the physical-chemical equilibrium (Eq. 1.9). If the logical variable Z_j is true, then there is no mass transfer in the j^{th} stage, and the streams flow through unchanged.

2. Challenges and scope of the thesis

Mathematical programming with MINLP formulation is a widespread approach in process synthesis.

Mainly the last two steps of mathematical programming, namely the formulation of the MINLP representation, and the solution of this model, are studied in the literature. The MINLP models are formulated in a way that it can be solved easily using one of the optimisation algorithms. Grossman (1996) defined three major guidelines of a „good" MINLP representation:

1. Keep the problem as linear as possible.
2. Develop a formulation whose NLP relaxation is as tight as possible.
3. If possible, reformulate the MINLP as a convex programming problem.

Multiplicity causes great problems in MINLP models. Multiplicity means that several solutions of the mathematical representation define the same structure. It can also be said that a structure is represented by isomorphic graphs. In this case, the objective function has the same value in several points. It makes more difficult finding the optimum, and the search space is unnecessarily big. Multiplicity is usually decreased in the second step of the mathematical programming, and the MINLP model is formulated in a way to have as low multiplicity as possible. However, in some cases, even the graph representation of the superstructure can be generated in a way to exclude isomorphic graphs.

An important characteristic of an MINLP model is the number of binary variables. With increasing number of binary variables, the complexity of the problem and the solution time increase exponentially. An MINLP model can be reformulated to decrease the number of binary variables by introducing new continuous variables and constraints. However, it would be more expedient to generate the graph representation of the superstructure in a way that the decrease of the number of binary variables is considered.

During my PhD study, I mainly dealt with the first step of the mathematical programming, the generation of the superstructure and its graph representation, and with the connection of this step to the other ones.

First I studied the use of earlier experience during the solution of a new synthesis problem. I applied a knowledge based method, case-based reasoning, in the generation of the proper superstructure with MINLP representation in distillation column synthesis.

Then I studied in details the connection between structures and graphs. I defined when an MINLP model represents a structure, and what the reasons of multiplicity are. I gave guidelines for how the multiplicity, and the number of binary variables of an MINLP representation, can be decreased in order to narrow the search space, to enhance the possibility of finding the global optimum, and to decrease the solution time.

Finally, I used my results and experience to develop a new superstructure and MINLP model for distillation column synthesis.

3. Case-based reasoning in mathematical programming

The first step of mathematical programming, generation of the superstructure, always requires engineering experience. Even in the combinatorial method of Friedler and co-workers (1992ab, 1993), the engineer has to select the set of units from which the superstructure will be generated. After generating the superstructure, the formulation of the mathematical model which exactly represents the superstructure is also a difficult task. Finally, in case of an MINLP model, the found optimum strongly depends on the initial point; therefore, it is worth to start the optimisation from a near optimal point.

To overcome these difficulties it seems evident to use the earlier experience in case of a new problem. For this aim, the best methods are perhaps the knowledge based methods. Among the knowledge based methods, I have chosen case-based reasoning to study the possibility its combination with mathematical programming because complex problems can be formulated easily in this method, and other techniques of process synthesis can be easily adopted and incorporated. In case-based reasoning, the most similar case to an actual problem is retrieved from a case library, and the solution of this case is used to solve the actual problem. Finally, the solution of the problem is stored in the case library for future use.

In this chapter I study the use of case-based reasoning for the generation of the superstructure in distillation column synthesis. The content of this chapter has been published in the paper Farkas et al. (2005a).

3.1. Problem statement

The problem addressed in this chapter is as follows. There is given an ideal mixture of components that is to be separated, by distillation, into a number of products of the specified composition. The goal is to propose a proper superstructure and an MINLP model to synthesize a sequence of distillation columns. The superstructure must include an initial structure for the design optimisation. As a realization of the data base of previously solved problems (a case library) has been created, and an efficient retrieval method which identifies the most similar case to the actual problem has been developed.

Pajula and co-workers (2001) developed a case-based reasoning method for the selection of single separations. Seuranen and co-workers (2005) further developed this approach for the synthesis of complex separation sequences.

According to my knowledge, this is the first case when case-based reasoning combined with

mathematical programming is used in distillation column synthesis. In order to simplify the study on the usability of case-based reasoning, the problem is restricted for ideal mixture separation cases. However, according to my opinion, the method can be further developed, for example to consider cases with azeotropic mixtures, as well.
The library of cases is built based on the detailed distillation examples with reproducible MINLP models published in the papers Aggarwal and Floudas, 1992; Viswanathan and Grossmann, 1993ab; Novak et al., 1996; Yeomans and Grossmann, 1999b; Caballero and Grossmann, 1999; Yeomans and Grossmann, 2000ab; and Caballero and Grossmann, 2001. The case library contains 26 cases of separation of ideal mixtures for up to five components.

The retrieval of the most similar cases to a new problem consists of the analysis of characteristics of feed, required products, operational parameters, and their comparison with the analogical properties of a problem under consideration.
As a solution, the superstructure and MINLP models of the most similar cases are suggested and theirs optimal flowsheets, which can be used as an initial point for optimisation.

3.2. Implementation of case-based reasoning method

The computer implementation of CBR is composed of three main parts: (1) case library; (2) retrieval method; and (3) adaptation.

3.2.1. Case representation

According to stated goal, a case must contain a superstructure with an applicable MINLP model, which can be used in design and optimisation to find the optimal structure and operational parameters. The case description is supplemented by a particular solution that can be used as an initial approximation in finding the optimal solution.

MINLP model with superstructure

The case library includes only cases with reproducible MINLP models. The representation of a model involves the superstructure, the set of variables and parameters, the mass and enthalpy balances and other constraints. However, usually only the superstructure, the variables and the main equations are detailed in the source articles, e.g. the equilibrium models and the basic mass balances are not represented. The articles contain the hints and

notes, which can help in using of a model. To provide the instructions for the use of MINLP model the original articles have been included in the case library as PDF files.

Solution of source cases

Usually the search space and the equations of MINLP models are strongly non-convex, so the identified optimum strongly depends on the starting point of the calculations. The solution of a similar problem is given as an initial in optimisation to increase greatly the probability to find the global optimum. The papers report usually a flowsheet supplemented with a dataset as a solution for a problem. The case library contains the flowsheet and their mathematical representation.

A flowsheet is represented as a graph. An example of graph representation of a flowsheet (Fig. 3.1, Yeomans and Grossmann, 1999b) is shown in Fig. 3.2. In this graph the nodes are the feed (F1), the distillation columns (C1, C2,…), the heat-exchangers (condensers: Con1, …; and reboilers: Reb1, …), the mixers/splitters (MS1, MS2, …) and the products (P1, P2, …); the edges are the flows between the units. This graph can be represented in matrix form (node-node matrix, Table 3.1). In this matrix a_{ij}=1 if there is connection from node i to node j, a_{ij}=0 otherwise.

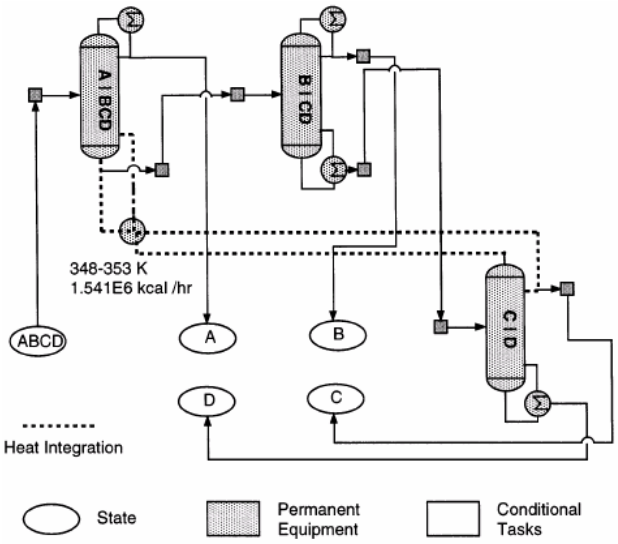


Figure 3.1. Example of flowsheet (*Fig. 5 in Yeomans and Grossmann, 1999b*)

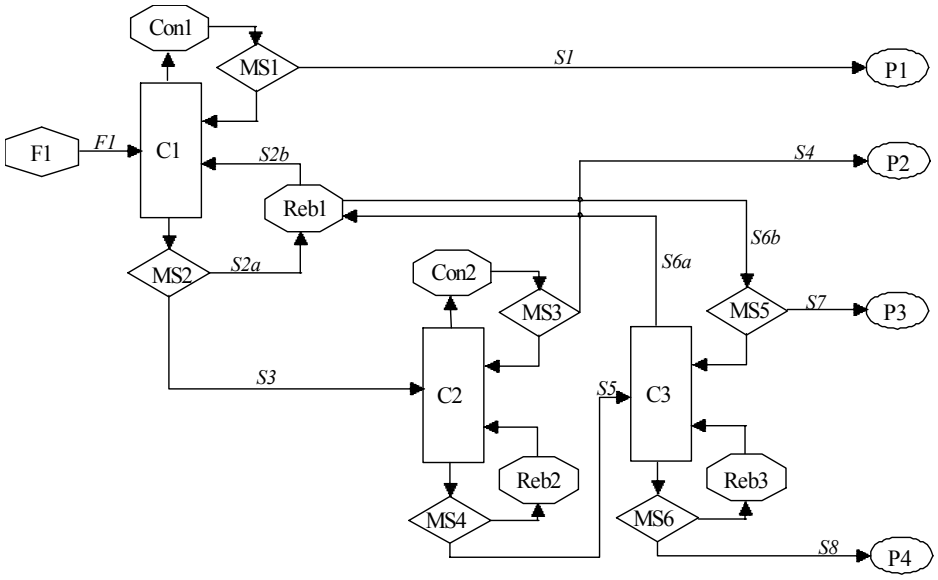


Figure 3.2. Graph representation of flowsheet

Many flows are supplemented with attributes such as temperature, flow rate, composition. Such flows have the captions (e.g. S1, S6b) in the graph. These flows are represented in separate edge-node matrix (Table 3.2), which contains the starting and ending nodes of the flows.

Table 3.1. Node-node matrix

	F1	C1	C2	C3	Con1	Con2	Reb1	Reb2	Reb3	MS1	MS2	MS3	MS4	MS5	MS6	P1	P2	P3	P4
F1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
C2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
C3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
Con1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Con2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Reb1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Reb2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reb3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
MS1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
MS2	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
MS3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
MS4	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
MS5	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
MS6	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

In the graph representation only simple columns are used, with maximum three inputs and two outputs. In case of thermally coupled flowsheets a possible rearrangement of the complex columns is used (see details in Example 3.1). If two flows between two columns have reverse direction then these flow pairs is called ”thermally coupled”. The thermally coupled complex columns are represented as composed of two parts – upper and lower separate columns.

Table 3.2. Edge-node matrix

	F1	S1	S2a	S2b	S3	S4	S5	S6a	S6b	S7	S8
start	F1	MS1	MS2	Reb1	MS2	MS3	MS4	C3	Reb1	MS5	MS6
end	C1	P1	Reb1	C1	C2	P2	C3	Reb1	MS5	P3	P4

The solution is represented by the graph (Fig. 3.2), the node-node matrix (Table 3.1) and the edge-node matrix (Table 3.2) as well as the detailed data of units and flows, such as:

distillation columns	- number of trays
	- diameter [m]
	- input/output trays
	- pressure [bar]
	- reflux ratio
heat exchangers	- area [m^2]
	- heat flowrate [MW]
	- utility
flows	- temperature [K]
	- flowrate [kmol/h]
	- set of components
	- mole fraction of components

In case of heat integrated columns the flows go through heat exchangers. Heat exchanger changes the temperature and physical condition of the flow. However, the rate of temperature changing is unknown. Therefore, these flows are marked with the same number, and distinguished with small letters (S2a, S2b,…), but only the data of the flow before heat exchanger are reported.

3.2.2. Case retrieval

During retrieval an actual problem is matched against previous ones from the case library, and the most similar problem is retrieved. The solution of the retrieved problem is used next in optimisation. The actual problem, which has to be solved, is the *target case*; and the solved problems with their solutions are the *source cases*.

First the case base is analyzed by induction method to classify the cases. One class of cases corresponding to the actual problem build retrieved set of cases. Next, the cases in the set are ranked according to their similarities to the target case using nearest neighbour method.

Inductive retrieval

Using a set of classification attributes the cases are grouped into clusters. The clusters are characterised by the following values of attributes:

Separation: sharp; non-sharp.

Heat integration: structure without heat integration; structure with heat integration; thermally coupled structure.

In the single column configuration only non-heat integrated structure is possible.

Number of products: 2; 3-5.

This attribute is considered because a model for single column configuration does not include the mass balances for the connection of distillation columns; thus, such model cannot be used for problem of separation of three or more products.

Feed type: single; multiple.

This attribute is considered because of the dissimilarity between the MINLP models with single and multiple feeds.

The cluster of cases corresponding to actual problem (that has the same values of classification attributes) is retrieved for next step – nearest neighbour method.

Nearest neighbour retrieval

The nearest neighbour method is used to calculate the similarity between the target case and the source cases from the set of similar cases retrieved by inductive method. The evaluation of the global similarity between the target and a source case is based on the computation of the local similarities. The local similarities deal with a single attribute, and take the value from the interval [0;1]. The global similarity can be derived from the local similarities as:

$$SIM^{T,S} = \frac{\sum_{a=1}^{n_a} w_a \cdot sim_a}{\sum_{a=1}^{n_a} w_a} \tag{3.1}$$

where w_a is the weight of importance of attribute a; sim_a is the local similarity between values of attribute a of the target (T) and the source case (S); n_a is the number of attributes. The weights of importance take integer value from 1 to 10 according to the actual requirements, where weight value 10 determines the most important attribute.

The similarity between component sets is very important and has to be applied first. It has to be determined which component in the source case corresponds to certain component in the target case. In simplest case, the sets of components of the target and the source case are identical. Otherwise the most similar sequence of components has to be determined, and identical components often do not create the corresponding pairs. For instance, the components set of the target case (Yeomans and Grossmann, 2000a) is *n*-butane; *n*-pentane; *n*-hexane. The components set of the source case (Yeomans and Grossmann, 2000b) is *n*-pentane; *n*-hexane; *n*-heptane. The *n*-pentane and *n*-hexane components are present in both cases, and it seems to be evident to assign them to each other in the target and in the source cases. Then the third pair of the components would be *n*-butane (target case) – *n*-heptane (source case). However, there is a problem with this assignment thanks to the fact that *n*-butane in the target case is the most volatile component, while *n*-heptane, the pair of *n*-butane in the source case, is the less volatile component. Thus, the solution of the source case cannot be used for the solution of the target case.

To overcome these difficulties, during matching of the components the primary assumption is the volatility order of the components, and the second is the nature of the components. The component pairs in previous example are *n*-butane – *n*-pentane; *n*-pentane – *n*-hexane; *n*-hexane – *n*-heptane. In this case the solution of the source case can be used to solve the target case.

In order to calculate the similarity five attributes are used: components, boiling points of components, molar masses of components, feed and product composition (mole fraction).

Components. It is a non-numeric attribute. The similarity of components is based on theirs chemical structure. The similarity tree, which includes all components in the case library (Fig. 3.3), has been built. In the similarity tree, the nodes represent the basic groups of chemical components. To each component group a numeric similarity value was assigned.

The similarity value of two components is the value of the nearest common node in the tree. For example when comparing *n*-butane and methanol the nearest common node is the 'organic' node, therefore the similarity value is 0.2. The more similar the components the greater is the similarity value between them. For the identical components the similarity value is 1.

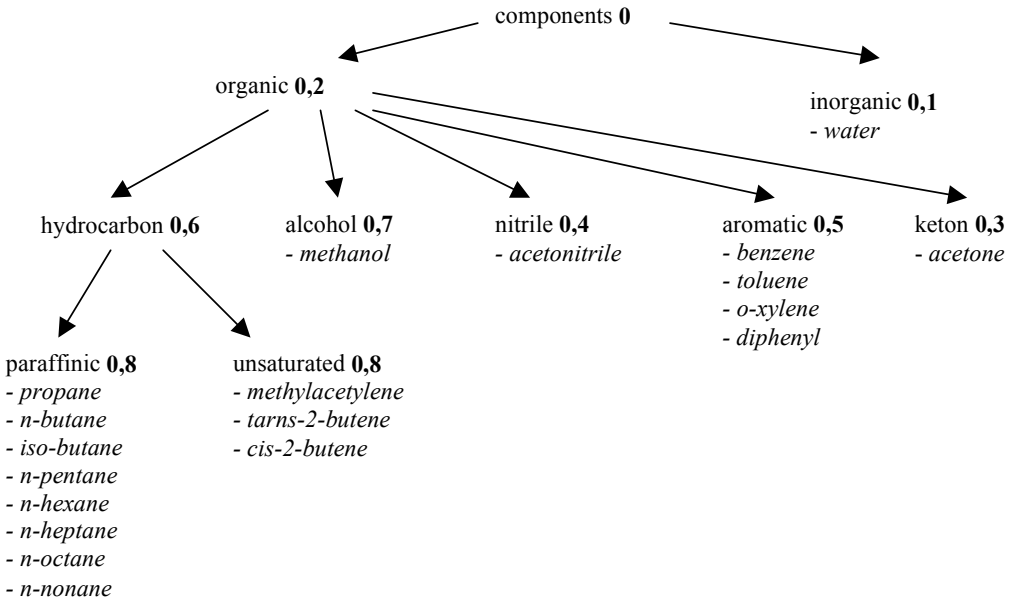


Figure 3.3. Similarity tree of components

It may happen that the cases with different numbers of products are compared. In this case there are components in one set, which have no corresponding components in another set. For these matchless components the nearest common node is the "components" node, therefore, the similarity value is 0 (see Example 3.2).

The local similarity of the components (sim_c) is defined as the average of the similarity values between the components:

$$sim_c = \frac{\sum_{i=1}^{n_c} sc_i}{n_c} \tag{3.2}$$

where sc_i is the similarity value of the components from the similarity tree; n_c is the maximal number of components in the compared mixtures.

As now only problems containing ideal mixtures are considered in the case library, this type of comparison of components, based only on the chemical structure of the components, is suitable. In the later phase of development also problems containing azeotropic mixtures

could be introduced to the case library, and the comparison of components can be further developed. Then the mixtures will be grouped according to what kind of and how many azeotropes are in the system, or the local similarity will be calculated based on a group contribution method.

Boiling point and molar mass of components. These attributes are numeric. In such case, the similarity of the attributes is calculated utilizing simple distance approach: the shorter a distance between two attribute's values the greater the similarity is. For the greater sensitivity not the original values are used, but normalized ones from interval [0;1]. The normalized values are defined for boiling point (t_b) and molar mass (m) as:

$$t_b = \frac{T_b - T_{b,min}}{T_{b,max} - T_{b,min}}, \tag{3.3}$$

$$m = \frac{M - M_{min}}{M_{max} - M_{min}}, \tag{3.4}$$

where $T_{b,min}$ is the smallest boiling point; $T_{b,max}$ is the highest boiling point; M_{min} is the smallest molar mass; M_{max} is the greatest molar mass in the case library.

The local similarities for these attributes are defined as:

$$sim_t = \frac{\sum_{i=1}^{n_c} \left(1 - \Delta t_{b,i}\right)}{n_c} \tag{3.5}$$

$$sim_m = \frac{\sum_{i=1}^{n_c} \left(1 - \Delta m_i\right)}{n_c} \tag{3.6}$$

where $\Delta t_{b,i}$ is the difference of the normalized boiling points; Δm_i is the difference of normalized molar masses; n_c is the maximal number of components.

In case of the different numbers of components of compared cases for matchless component the difference of boiling points ($\Delta t_{b,i}$) or molar masses (Δm_i) is the matchless component's normalized boiling point or normalized molar mass (see more Example 3.2).

Feed and product compositions. These are also numeric attributes that are vectors. Comparing vector attributes the length of distance vector **d**, is determined.

$$\begin{gathered}\mathbf{T} = \left(a_1^T, a_2^T, ..., a_{n_c}^T\right), \quad \mathbf{S} = \left(a_1^S, a_2^S, ..., a_{n_c}^S\right), \quad a \in [0;1]; \\ |\mathbf{d}| = \sqrt{\left(a_1^T - a_1^S\right)^2 + \left(a_2^T - a_2^S\right)^2 + ... + \left(a_{n_c}^T - a_{n_c}^S\right)^2} \quad \mathbf{d} \in \mathbb{R}^{n_c}\end{gathered} \tag{3.7}$$

where **T** is the attribute vector of the target case; **S** is the attribute vector of the source case.

In case of the different numbers of components of compared cases zero elements are added to

the shorter vector in order to have the same number of elements in the compared vectors (see more Example 3.2).

Because there are a number of product composition vectors, the difference vector and the distance are calculated for every product pair. The method is analogical for the problems with multiple feeds. The local similarity of feed compositions (sim_f) and product compositions (sim_p) are defined as:

$$sim_f = 1 - \frac{1}{n_f} \sum_{j=1}^{n_f} \frac{\left| \mathbf{d}_{f,j} \right|}{\left| \sum_{i=1}^{n_c} \mathbf{e}_i \right|} \qquad \mathbf{d}_{f,j}, \mathbf{e}_i \in \mathbb{R}^{n_c} \tag{3.8}$$

$$sim_p = 1 - \frac{1}{n_p} \sum_{j=1}^{n_p} \frac{\left| \mathbf{d}_{p,j} \right|}{\left| \sum_{i=1}^{n_c} \mathbf{e}_i \right|} \qquad \mathbf{d}_{p,j}, \mathbf{e}_i \in \mathbb{R}^{n_c} \tag{3.9}$$

where n_f is the number of feeds; n_p is the number of products; $\mathbf{e}_i$ are the basis vectors in the $\mathbb{R}^{n_c}$ space (necessary for normalization).

Other attributes can also be considered according to the actual requirements. The calculation of similarity for other numeric or vector values is performed in the same way.

Using the nearest neighbour method the cases of the set, retrieved by inductive method, are ranked, and the solution of the most similar case is found. The MINLP model with the superstructure and the optimal solution of the source case are suggested. Usually the chosen solution has to be adapted in order to meet the actual requirements.

3.2.3. Adaptation

The three most similar cases are reported as potential solutions, and according to the actual requirements and engineering experience the most useful model is chosen. Because of the complexity of the distillation problems there is no automatic adaptation of the found solution. The task of the designer is the modification of the MINLP model and the reuse of the solution of the chosen case as an initial point for design and optimisation.

3.3. Examples

In this section three examples are presented to illustrate the mathematical representation of a solution of a case, the retrieval method and problem solving. Example 3.1 shows the

mathematical representation of a thermally coupled structure, how to rearrange the configuration of complex distillation columns. Example 3.2 illustrates the retrieval, the comparison of a target and a source case. The nearest neighbour method is applied to the case with the different number of components and feeds. The solving of a new problem is presented in Example 3.3.

3.3.1. Example 3.1.

As it was mentioned in subsection 3.2.1, a flowsheet is represented as a graph. The nodes of the graph are the units (columns, condensers, reboilers and mixers/splitters), the feeds and the products; the edges are the streams connecting the units. This graph is represented also with a node-node matrix and an edge-node matrix.

Only simple distillation columns with maximum three input and two output streams can be used in the graph. In the case of thermally coupled solution a possible rearrangement of the flowsheet could also be represented in the graph. An example for this rearrangement is shown below.

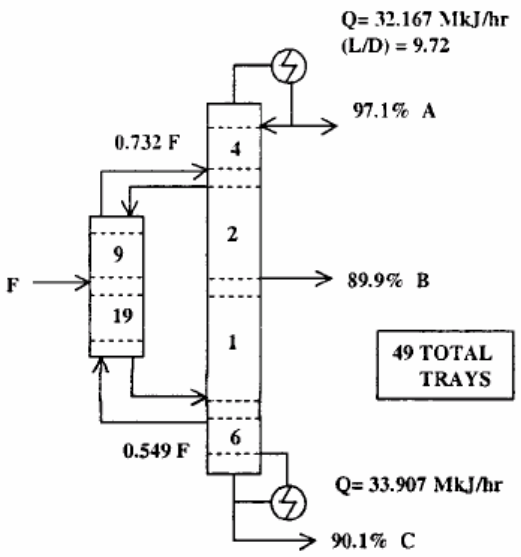


Figure 3.4a. Flowsheet of Example 3.1 (*Fig. 5e in Yeomans and Grossmann, 2000b*)

The examined flowsheet (Yeomans and Grossmann, 2000b) is presented in Fig. 3.4a. There are two columns, a simple column (without heat exchanger) and a complex column. The simple column meets our requirements of the mathematical graph representation. However, the complex column has three outputs, therefore, it has to be splitted into two simple columns (Fig. 3.4b). Between these two columns there are interconnection streams in both directions,

but one of these streams is broken with a splitter. The node-node matrix and the edge-node matrix of the graph are given in Table 3.3 and Table 3.4.

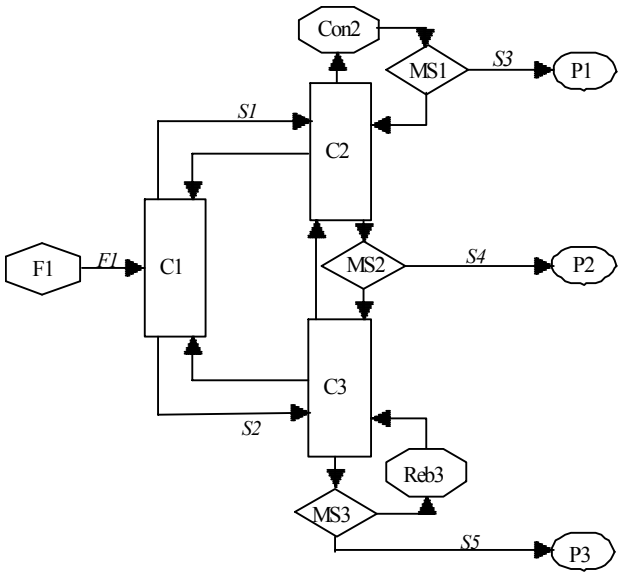


Figure 3.4b. Graph representation of flowsheet of Example 3.1

Table 3.3. Node-node matrix of Example 3.1

	F1	C1	C2	C3	Con2	Reb3	MS1	MS2	MS3	P1	P2	P3
F1	0	1	0	0	0	0	0	0	0	0	0	0
C1	0	0	1	1	0	0	0	1	0	0	0	0
C2	0	1	0	0	1	0	0	1	0	0	0	0
C3	0	1	1	0	0	0	0	0	1	0	0	0
Con2	0	0	0	0	0	0	1	0	0	0	0	0
Reb3	0	0	0	1	0	0	0	0	0	0	0	0
MS1	0	0	1	0	0	0	0	0	0	1	0	0
MS2	0	0	0	1	0	0	0	0	0	0	1	0
MS3	0	0	0	0	0	1	0	0	0	0	0	1
P1	0	0	0	0	0	0	0	0	0	0	0	0
P2	0	0	0	0	0	0	0	0	0	0	0	0
P3	0	0	0	0	0	0	0	0	0	0	0	0

Between the first column (C1) and the second column (C2) there are two streams (the situation is the same between the first and the third column). These streams can be regarded as outputs/inputs from or to the column, but also as a reflux in the first column. These kinds of streams, which have neighbour of reverse direction, are called *thermally coupled streams*. In Fig. 3.4b, the marked streams S1 and S2 are examples of such streams (see Table 3.6). The S1 stream is regarded as an output from the first column, and an input to the second column (see Table 3.5).

Table 3.4. Edge-node matrix of Example 3.1

	F1	S1	S2	S3	S4	S5
start	F1	C1	C1	MS1	MS2	MS3
end	C1	C2	C3	P1	P2	P3

The complex column is splitted into two simple columns connected by *thermally coupled streams*. One of these streams is broken with a splitter, from which a product stream (S4) starts. This stream can be regarded as a product of a second and the third column. In the graph representation such a stream is regarded as the bottom product of the column (see Table 3.5).

Table 3.5. Units of Example 3.1

	Column 1.	Column 2.	Column 3.
number of trays	31	9	9
feed	F1	S1	S2
feed tray*	21	4	8
top product	S1	S3	-
bottom product	S2	S4	S5
reflux ratio		9.72	
condenser utility	-	cold utility	-
condenser heat flowrate (MW)	-	32.167	-
reboiler utility	-	-	hot utility
reboiler heat flowrate (MW)	-	-	33.907

* trays are counted from bottom up

Table 3.6. Streams of Example 3.1

	Main component(s)	Flowrate (kmol/h)	Composition	Type
F1	ABC	1000	(0.20; 0.50; 0.30)	normal
S1		732		therm. coupl.
S2		549		therm. coupl.
S3	A		(0.971; -; -)	normal
S4	B		(-; 0.899; -)	normal
S5	C		(-; -; 0.901)	normal

In the case of thermally coupled flowsheets the complex columns are separated into simple columns with maximum two outputs. The internal streams of the earlier complex column are always regarded as outputs of the columns above the actual stream. The pair of streams with reverse directions from one column to another are marked as *thermally coupled streams*. However, these streams are regarded as normal streams.

3.3.2. Example 3.2

To illustrate the retrieval method only the comparison of two cases are described.

The target case (Viswanathan and Grossmann, 1993a) is a benzene, toluene, *o*-xylene system, the source case (Viswanathan and Grossmann, 1993a) is a methanol, water system. The descriptions of the cases are presented in Table 3.7.

The target case is sharp separation without heat integration; there are two products and multiple feed. The source case has the same values of classification attributes, hence, it belongs to the cluster of the target case.

Table 3.7. Descriptions of target and source cases of Example 3.2

	Target case	Source case
system	benzene–toluene–*o*-xylene	methanol–water
condenser type	total	total
reboiler type	kettle type	kettle type
estimated maximum number of trays	40	60
feed 1	$Feed_1^T$=50. $\mathbf{x}_1^{F,T}$=(0.15; 0.25; 0.60) $P_1^{F,T}$=1.2 bar. $T_1^{F,T}$=411.459 K. $q_1^{F,T}$=0.1	$Feed_1^S$=43.5. $\mathbf{x}_1^{F,S}$=(0.15; 0.85) $P_1^{F,S}$=1.42 bar. $T_1^{F,S}$=365.0 K. $q_1^{F,S}$=0.0
feed 2	$Feed_2^T$=50. $\mathbf{x}_2^{F,T}$=(0.55; 0.25; 0.20) $P_2^{F,T}$=1.2 bar. $T_2^{F,T}$=390.387 K. $q_2^{F,T}$=0.0	$Feed_2^S$=29.5. $\mathbf{x}_2^{F,S}$=(0.50; 0.50) $P_2^{F,S}$=4.8 bar. $T_2^{F,S}$=392.697 K. $q_2^{F,S}$=0.0
feed 3	–	$Feed_3^S$=27.0. $\mathbf{x}_3^{F,S}$=(0.89; 0.11) $P_3^{F,S}$=1.38 bar. $T_3^{F,S}$=347.797 K. $q_3^{F,S}$=0.0
pressures	$P^{R,T}$=1.2 bar. $P^{B,T}$=1.20 bar. $P^{D,T}$=1.10 bar. $P^{C,T}$=1.05 bar	$P^{R,S}$=1.4475 bar. $P^{B,S}$=1.44064 bar. $P^{D,S}$=1.0408 bar. $P^{C,S}$=1.0340 bar
purity constraint on top product	$x_{45,1}^T \geq 0.999$	$x_{60,1}^S \geq 0.999$
purity constraint in bottom product	$x_{1,2}^T + x_{1,3}^T \geq 0.999$	$x_{1,2}^S \geq 0.999$
upper bound of reflux ratio	2	20

Before the use of nearest neighbour formula, the corresponding pairs of components have to be identified. The primary assumption is the volatility order of the components. The data of

the components are given in Table 3.8. According to the boiling points the most similar component to benzene is methanol in the source case, and the second component pair is the toluene – water. The *o*-xylene has no pair, because there are different numbers of products in the target and the source cases.

Comparing the identity of components similarity for the benzene–methanol pair the nearest common node in the similarity tree (Fig. 3.3) is the 'organic' node (sc_1=0,2), for the toluene–water pair is the 'components' node (sc_2=0). The *o*-xylene has no pair, therefore, for this component the nearest node is the 'components' node, the similarity value is sc_3=0. The local similarity value of components (sim_c) is the average of those similarity assessments (Table 3.9).

Table 3.8. Boiling and molar masses of the components of Example 3.2

	T_b, boiling point (K)	t_b, normalized boiling point	M, molar mass (g/mol)	m, normalized molar mass
benzene	353.2	0.371	78	0.472
toluene	383.8	0.481	92	0.583
o-xylene	417.6	0.603	106	0.693
methanol	337.9	0.316	32	0.110
water	373.15	0.443	18	0.000

Table 3.9. Comparison of boling point and molar mass of components of Example 3.2

	sc_i	$\Delta t_{b,i}$ (K)	Δm_i (g/mol)
benzene–methanol	0.2	0.055	0.362
toluene–water	0	0.038	0.583
o-xylene	0	0.603	0.693
local similarity values	sim_c=0.067	sim_t=0.232	sim_m=0.546

To determine the local similarity values of boiling points and molar masses first the differences of the component pairs are determined. In the case of *o*-xylene, which has no pair from the source case, the differences are its own normalized values ($\Delta t_{b,i}=t_{b,i}$; $\Delta m_i=m_i$). The local similarity values are the average of the differences of component pairs (Table 3.9).

The purity requirements are not considered when comparing the products but the concentrations of the components in the required products. It is necessary because in non-sharp separations there are no purity requirements, the exact composition of the products are used instead. Therefore, the required product compositions have to be given also for the sharp separations. For example in the target case, only the minimum mole fraction of the benzene of the top product is known ($x^T_{45,1} \geq 0.999$), therefore the concentration of the components in the

required product is $x_{p,1}^T$=(0.999; 0; 0). This is not really product composition because it does not fulfil the requirement that the sum of mole fractions is 1. It is rather the minimal mole fraction of *each* component in the product. Therefore, the compared products are:

$$\begin{aligned} &\mathbf{x}_{p,1}^T=(0.999;\ 0;\ 0) \quad - \quad \mathbf{x}_{p,1}^S=(0.999;\ 0;\ 0) \\ &\mathbf{x}_{p,2}^T=(0;\ 0;\ 0) \quad - \quad \mathbf{x}_{p,2}^S=(0;\ 0.999;\ 0) \end{aligned} \tag{3.10}$$

In the second product of the target case all the mole fractions are zero, because the constraint $x_{1,2}^T + x_{1,3}^T \geq 0.999$ is uncertain. The third zero mole fraction is added to the product vector of the source case in order to have the same number of elements of compared vectors.

The products are matched by minimizing the difference between the product component vectors. Therefore, in this case the product pairs are $\mathbf{x}_{p,1}^T - \mathbf{x}_{p,1}^S$ and $\mathbf{x}_{p,2}^T - \mathbf{x}_{p,2}^S$. For every product pair the difference between the product composition vectors is calculated:

$$\begin{aligned} &\mathbf{x}_{p,1}^T = (0.999;0;0) \quad \mathbf{x}_{p,1}^S = (0.999;0;0) \quad \left|\mathbf{d}_{p,1}\right| = \sqrt{\left(0^2+0^2+0^2\right)} = 0 \\ &\mathbf{x}_{p,2}^T = (0;0;0) \quad \mathbf{x}_{p,2}^S = (0;0.999;0) \quad \left|\mathbf{d}_{p,2}\right| = \sqrt{\left(0^2+0.999^2+0^2\right)} = 0.999 \end{aligned} \tag{3.11}$$

The local similarity value of product compositions is:

$$sim_p = 1 - \frac{\sum_{j=1}^{2}\left|\mathbf{d}_{p,j}\right|}{2\cdot\left|\sum_{i=1}^{3}\mathbf{e}_i\right|} = 1 - \frac{0+0.999}{2\cdot\sqrt{3}} \approx 0.712 \tag{3.12}$$

The similarity of the feed compositions is calculated analogically as for the product ones. In order to have the same number of elements of vectors, a third zero element is added to the feed compositions of the source case. The number of feeds in source cases is greater than in the target case, therefore, a third feed with zero elements is considered in the target case. When matching the feed also the primary objective is the minimal difference, thus, the feed pairs, and the distance between the feed composition vectors are:

$$\begin{aligned} &\mathbf{x}_{f,1}^T = (0.15;0.25;0.60) \quad \mathbf{x}_{f,1}^S = (0.15;0.85;0) \quad \left|\mathbf{d}_{f,1}\right| = \sqrt{\left(0^2+0.6^2+0.6^2\right)} \approx 0.849 \\ &\mathbf{x}_{f,2}^T = (0.55;0.25;0.20) \quad \mathbf{x}_{f,3}^S = (0.89;0.11;0) \quad \left|\mathbf{d}_{f,2}\right| = \sqrt{\left(0.44^2+0.14^2+0.2^2\right)} \approx 0.503 \\ &\mathbf{x}_{f,3}^T = (0;0;0) \qquad\qquad \mathbf{x}_{f,2}^S = (0.50;0.50;0) \quad \left|\mathbf{d}_{f,3}\right| = \sqrt{\left(0.5^2+0.5^2+0^2\right)} \approx 0.707 \end{aligned} \tag{3.13}$$

The local similarity value of feed compositions is:

$$sim_f = 1 - \frac{\sum_{j=1}^{3} \left| \mathbf{d}_{f,j} \right|}{3 \cdot \left| \sum_{i=1}^{3} \mathbf{e}_i \right|} \approx 1 - \frac{0.849 + 0.503 + 0.707}{3 \cdot \sqrt{3}} \approx 0.604 \tag{3.14}$$

The weights of importance for local similarities have to be assigned before using Eq. 3.1. The weights have values from 1 to 10. The more important an attribute the greater the value of the weight is. A possible set of weights is shown in Table 3.10.

Table 3.10. The weights of importance for Example 3.2

Attributes	Weights
Quality of components	10
Composition of required products	7
Boiling point of components	4
Composition of feeds	3
Molar mass of components	1

The global similarity is calculated as follows:

$$SIM^{T,S} = \frac{w_c \cdot sim_c + w_t \cdot sim_t + w_m \cdot sim_m + w_p \cdot sim_p + w_f \cdot sim_f}{\sum_{i=1}^{5} w_i} = \tag{3.15}$$

$$= \frac{10 \cdot 0.067 + 4 \cdot 0.232 + 1 \cdot 0.546 + 7 \cdot 0.712 + 3 \cdot 0.604}{25} = 0.467$$

In the above example the comparison of two cases has been presented. First, the components were matched according to the volatility order and the boiling points. The local similarity of the components was calculated applying the values obtained from the similarity tree. The distance between the boiling point and the molar mass of the components were calculated; for the matchless component the differences were its normalized values. During the determination of the local similarities of products and feeds a third zero element was added in the source case compositions, in order to have the same number of elements of vectors, and a third feed was considered as **0** vector to the target case, in order to have the same number of feeds. A set of importance weights was assessed, and the global similarity was calculated.

3.3.3. Example 3.3

Third example presents the application of the method to a new distillation problem.

There is given a heptane-toluene mixture. The flowrate of the equimolar [0.5, 0.5] feed is 100 kmol/h. The target is to separate the mixture into pure components with 95% purity requirement at the top and at the bottom.

It is a sharp separation problem and single column configuration should be used. It means that the searched structure is not heat integrated. There are one feed and two products. Applying the inductive retrieval, the set composed of four source cases has been determined in the case library. Next, the global similarity is calculated for the target case and for all the source cases using the nearest neighbour method. As a result, the product compositions of the target case is [0.95, 0] at the top, and [0, 0.95] at the bottom. When it is required, a zero element is added to the composition vector.

Table 3.11. Nearest neighbour retrieval (Example 3.3)

	source case 1	source case 2	source case 3	source case 4
problem published originally	Viswanathan and Grossmann (1993a) Example Ternary 1	Viswanathan and Grossmann (1993a) Example Ternary 2	Viswanathan and Grossmann (1993a) Example Unit	Yeomans and Grossmann (2000a) Example 1
system	benzene toluene *o*-xylene	benzene toluene *o*-xylene	acetone acetonitrile water	benzene toluene
sim_c	0.400	0.400	0.133	0.600
sim_t	0.777	0.777	0.767	0.967
sim_m	0.711	0.711	0.756	0.913
sim_p	0.329	0.713	0.714	0.650
sim_f	0.822	0.822	0.714	0.833
SIM	0.503	0.611	0.492	**0.713**

According to the nearest neighbour method (see Table 3.11) the most similar case is a benzene-toluene problem (Yeomans and Grossmann, 2000a). In the design and optimisation of the target case the superstructure and the MINLP model of this most similar case is suggested to use. Here the use of the first and the second case is shown. As the third case uses the same MINLP model as the second one (Viswanathan and Grossmann, 1993a) during the optimisation only the initial point would be different. Therefore, the use of the third case is not studied here.

The models have to be adapted according to the actual requirements of the target case. The adaptation has two main steps: (1) adaptation of the model; and (2) adaptation of the solutions of the source cases as initial point.

The adaptation of the MINLP model is based on the assumptions used during the optimisation. The column pressure assumed to be constant; therefore, the equations of the pressure profile in the model of Viswanathan and Grossmann (1993a) are omitted. Constant molar overflow is assumed; therefore, the enthalpy balances and enthalpy calculations are omitted, and other equations are used instead which force the total vapor and liquid flows to be constant in each column section. The fugacities are calculated according to the Raoult-Dalton equation:

$$f_c^V = P \cdot y_c , \tag{3.16}$$

$$f_c^L = x_c \cdot p_c^0(T) , \tag{3.17}$$

where f_c^V is the vapor fugacity [Pa]; P is column pressure [Pa]; y_c is vapor mole fraction [-]; f_c^L is the liquid fugacity [Pa]; x_c is liquid vapor mole fraction [-]; and $p_c^0(T)$ is vapor pressure [Pa], all belonging to component c. Vapor pressure $p_c^0(T)$ is calculated with Antoine equation:

$$\log\left(p_c^0[\text{Hgmm}]\right) = A_c - \frac{B_c}{C_c + T[^\circ\text{C}]} , \tag{3.18}$$

where A_c, B_c and C_c are the Antoine constants of component c. The applied model parameters (Gmehling et al., 1977) are collected in Table A.1 in the Appendix.

As the heptane – toluene mixture has lower relative volatility than the mixtures of the source cases, the maximum number of trays in the column is increased to 80.

According to our earlier experiences the numerical characteristics of this kind of models can be improved by adding monotony constraints to the model. Therefore, concentration and temperature monotony constraints are given to the MINLP models which do not spoil the generality of the models.

The cost function is also modified according to the actual requirements. The following cost function is applied (Luyben and Floudas, 1994):

$$TAC = \beta_{tax}(c_{LPS} \cdot \Delta H_{vap} + c_{CW} \cdot \Delta H_{cond})V + \mathrm{f}(N_{st}, DC) / \beta_{pay} \tag{3.19}$$

$$\mathrm{f}(N_{st}, DC) = 12.3[615 + 324DC^2 + 486(6 + 0.76N_{st})DC] + 245N_{st}(0.7 + 1.5DC^2) \tag{3.20}$$

where β_{tax} is the tax factor (=1.18); c_{LPS} is the cost of the low pressure steam (= $3.54 \cdot 10^{-4}$ USD/kJ); c_{CW} is the cost of the cooling water (= $2 \cdot 10^{-7}$ USD/kJ); ΔH_{vap} is the latent heat of vaporization (= 33773 kJ/kmol); ΔH_{cond} is the latent heat of condensation (= 31828 kJ/kmol); V is the vapor flowrate at the bottom [kmol/yr]; β_{pay} is the payback period (= 15 yr); N_{st} is the number of equilibrium stages in the column; DC is the column diameter [m].

The column diameter is calculated from the cross section of the column (A, [m^2]):

$$DC = 2\sqrt{\frac{A}{\pi}} \tag{3.21}$$

The cross section of the column is determined by the flowrate of the vapor stream and the density of the vapor in the reboiler, using the F_f–procession (Kister, 1992):

$$A = \frac{\dot{m}_V}{F_{max}\sqrt{\rho_V}} \tag{3.22}$$

where $\dot{m}_V$ is the amount flowrate of the vapor [kg/s]; F_{max} is the F-factor ($=2.2\sqrt{Pa}$); and ρ_V is the density of the vapor [kg/m^3].

The solution of the source case is used to give initial state in the design and optimisation. In the solution of the most similar source case (Yeomans and Grossmann, 2000a) the number of trays was 55, the reflux ratio was 1.77, and the column diameter was 0.56 m (=1.84 ft). As in our case the feed is different from the feed of the source case (100 kmol/h instead of 150 kmol/h), the values of these quantities are modified in the initial state. Because of the lower relative volatility of our mixture the reflux ratio and the column diameter are increased to double (3.54 and 1.12 m, respectively) using the same number of trays (55) in the initial state. An initial column profile is calculated dividing the mole fraction interval between the purities of distillate and bottom product into the same number of intervals as the number of trays. The initial temperature profile of the column is calculated similarly. The initial value of all the other variables is calculated from these initial values using the model equations.

In the solution of the second most similar source case (Viswanathan and Grossmann, 1993b) the number of trays was 25, the reflux ratio was 9.01, the flowrate of the distillate was 15 kmol/h, and the flowrate of the bottom product was 85 kmol/h. In this solution low number of trays is used with very high reflux ratio, therefore, in the initial state of the new problem the number of trays is double (50), and the reflux ratio is half (4.50) as in the solution of the source case. In our problem the purity requirements in the distillate and in the bottom product for the main component are the same, therefore, the initial value of the distillate and the bottom product are the same, 50 kmol/h. An initial column profile for concentrations and temperature is calculated by the same way as in the first case.

The adapted MINLP model is solved by GAMS DICOPT++ (Brooke et al., 1992) on a Sun Sparc Station. The results of both MINLP models are presented in Table 3.12.

Table 3.12. Results of Example 3.3

Model	Yeomans and Grossmann (2000a)	Viswanathan and Grossmann (1993a)
objective (USD/yr)	3 066 973	3 121 234
number of trays	63	73
feed location (from bottom)	30	38
column diameter [m]	1.51	1.50
reflux ratio	4.22	4.18

The results are checked with ChemCAD 5.2 simulator fixing the parameters given in Table 3.12. For the solution of the model of Yeomans and Grossmann (2000a) the purity of the distillate is 0.9408, of the bottom product is 0.9444. For the solution of the model of Viswanathan and Grossmann (1993a) the purity of both products are 0.9407.

It can be seen, that in this case the MINLP model of the most similar source case found better solution than the model of the second most similar source case.

3.4. Summary

In this chapter the application of case-based reasoning method is presented for finding superstructure with an MINLP model, and a solution of the corresponding distillation synthesis problem by suggesting an initial point for performing design and optimisation of the system.

The cases in the case library are earlier published distillation problems with reproducible MINLP models. Each case contains a problem description, and the mathematical representation of its solution.

When solving a target (new) problem, the most similar case to the target is found in the case library during the retrieval process. First, a set of matching cases is retrieved using inductive retrieval. The cases are classified according to the operational attributes like sharp/non-sharp separation, heat integration, number of products and feeds. Then the cases in the retrieved set are ranked according to their similarity to the target case using the nearest neighbour method. In the nearest neighbour retrieval, the similarity is calculated from the local similarities of components, their molar masses and boiling points, and composition of feeds and required products. Weights are set for the specific problem.

After the retrieval, the three most similar cases are presented to the system user. The corresponding MINLP model can be generated based on the original articles, referring the

selected most similar cases, and the optimal solution of these cases can be used as starting point for design and optimisation.

The developed method shows that a knowledge based method can be used in process synthesis for the preparation of mathematical programming model formulation, namely in the generation of the superstructure.

4. Graph representations and mathematical models

Once a superstructure is generated, its graph representation, and the mathematical model of the graph representation, can be formulated. An expert engineer may have the ability to perform this process, and to generate an exact and mistake-free mathematical model, which will be optimised. However, this is a very difficult and complicated task, and needs a systematic procedure.

If the mathematical model is generated, it is usually not guaranteed that it really represents the superstructure. A model represents a superstructure if each solution of the model is in accordance with a structure and to the design and operational parameters of the structure. The check of this correspondence is not evident, and it has not been studied until now.

The mathematical model is usually improved to be easier to solve using a proper solver or algorithm. Grossmann (1996) defined three major guidelines of a „good" MINLP representation:

1. Keep the problem as linear as possible.
2. Develop a formulation whose NLP relaxation is as tight as possible.
3. If possible, reformulate the MINLP as a convex programming problem.

The check of these criterion needs deep mathematical insight to the model. A mathematical model can also be improved simply by decreasing the multiplicity (i.e. excluding the solutions representing the same structure), and the number of logical/binary variables. In this way the search space of the problem can be really tighten and, therefore, the solution time can be decreased, and the chance of finding the global optimum can be increased drastically.

In this chapter a method is presented for automatic generation of the mathematical representation of a superstructure. The automatically generated MINLP model can be used as a reference for other models; that is, it can be checked whether a model represents the superstructure, or not. Finally, guidelines are given to how the multiplicity of the model, and the number of binary variables can be decreased.

The content of this chapter has been published in papers Farkas et al. (2005bc).

4.1. Relations between structures and graphs

In a process synthesis procedure usually infinite number of physically feasible process structures can be used to achieve the targeted chemical process, although most of them are far from being optimal. For applying MINLP technique with superstructure approach, the

engineer have to select from the physically feasible structures a finite set of process structures to be studied. These preliminary selected process structures will here be called *considered structures*. This set of considered structures can be assigned by any explicit or implicit way.
A superstructure may contain structures that are not even physically feasible; these are *infeasible structures*. Those feasible structures that are not considered, together with the physically infeasible structures are simply called *non-considered structures*. The system of subsets in the set of substructures of the superstructure is illustrated in Fig. 4.1.

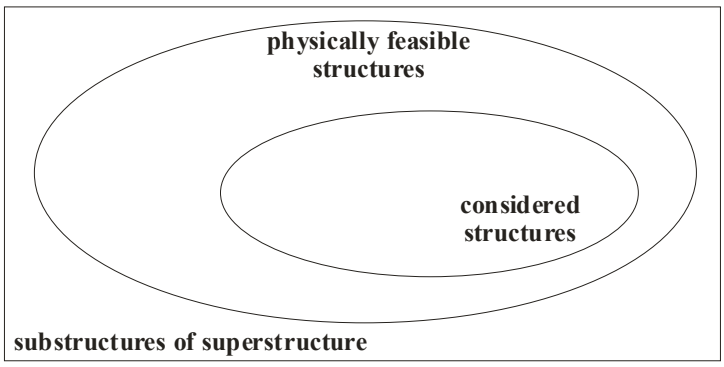


Figure 4.1. Classification of structures

Once this set of considered structures is given, the engineer has to construct a mathematically treatable superstructure that includes all the considered structures as its substructures. This superstructure itself is usually represented by a graph or network and/or by a system of variables and mathematical relations.
In the present dissertation the R-graph representation of structures will be used. From here on the simple term graph will be applied instead of R-graph.
The graph representation of the superstructure will here be called the *supergraph*.

As it was mentioned in the literature review, structural multiplicity is an important phenomenon caused by the possibility of representing the same structure with different graphs. For example the two subgraphs (Fig. 4.2b) of the supergraph in Fig. 4.2a represent the same structure, because the types of Unit 2 and Unit 3 are identical (Type B). It means, that these subgraphs are isomorphic graphs.

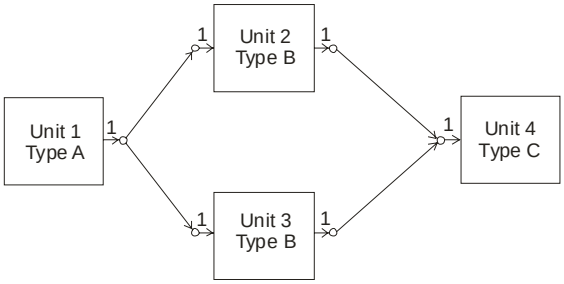


Figure 4.2a. Graph of a superstructure

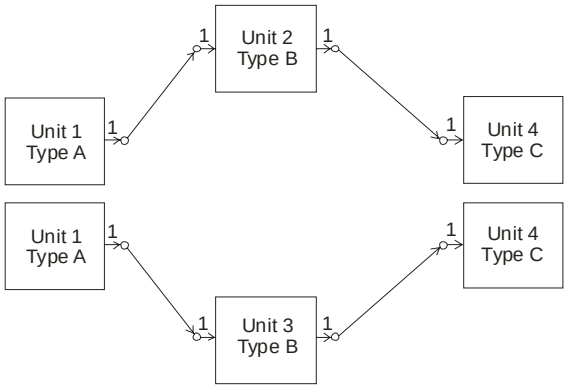


Figure 4.2b. Isomorphic subgraphs representing the same structure

Structural multiplicity of a structure s in a superstructure s^* represented by a graph r^* is defined as the number of all those subgraphs of r^* that represent s. In the example above, the multiplicity of the structure represented by the graphs shown in Fig. 4.2b is 2 inside the superstructure represented by the graph shown in Fig. 4.2a.
On the other hand, *redundancy of a structure* in itself is defined as the difference between the number of all the subgraphs of a graph representing the structure and the number of all the substructures of the studied structure.

As outlined above, the set of the substructures of a superstructure is classified according to whether the structure is physically feasible, and if yes then whether it is considered by the engineer. The same distinction can be made amongst the graphs representing the structures.
Any graph representing physically feasible structure is, naturally, called *feasible graph*. All the other graphs (if they are anyhow constructed) are represented by so-called *infeasible graphs*. The relation between considered structures and their representing graphs is just a bit more complicated because considered structures may be represented by considered graphs and non-considered graphs as well.

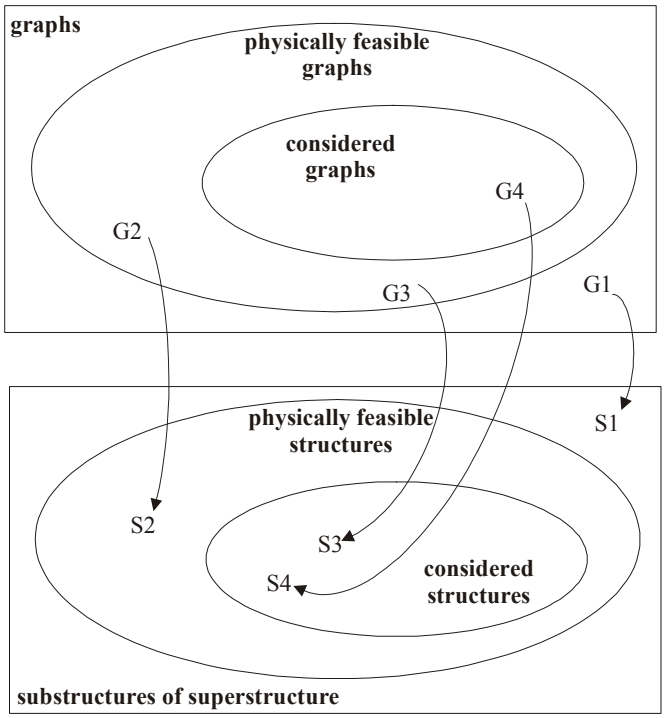


Figure 4.3. Relations between feasible and considered structures and graphs

Since the detrimental effect of structural multiplicity and redundancy can be decreased by non-considering isomorphic graphs representing the same structure, the set of *considered graphs* is defined to be such a set of graphs that each considered structure is represented by exactly one graph in this set, and each element of this set represents a considered structure. It follows, that there are no isomorphic graphs in the set of considered graphs.
The relations between the mentioned sets are shown in Fig. 4.3.

4.2. Example 4.1 – Problem statement

The notions explicated in section 4.1 are demonstrated in this small example. A small planning problem (Kocis and Grossmann, 1987; Raman and Grossmann, 1992) is considered to demonstrate both the new concepts and how the methodology works. The process is to produce product C from raw materials A and/or B with maximal profit. Three units can be used to accomplish this aim, as it is shown in Fig. 4.4. Data are given in Table 4.1.

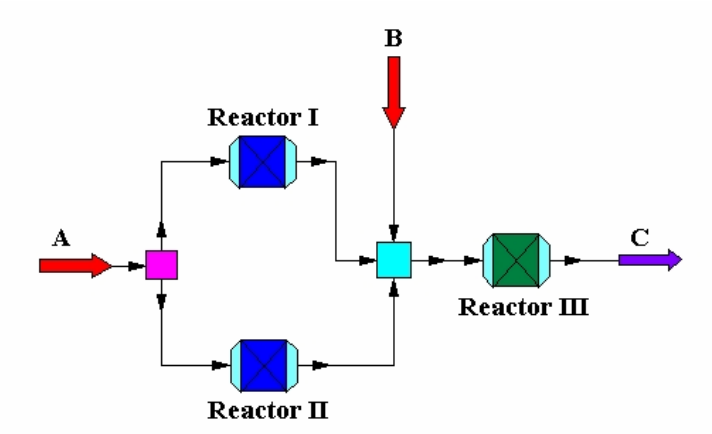


Figure 4.4. Example superstructure

Table 4.2. Data for Example 4.1

unit	Fixed cost 10^3 USD/h	Variable cost 10^3 USD/ton of product
I	1.0	1.0
II	1.5	1.2
III	3.5	2.0
raw material costs, 10^3 USD		1.8/ton of A, 7.0/ton of B
revenue, 10^3 USD		13.0/ton of C
Mass balances for units		
unit I: $b_2 = \ln(1+a_2)$ unit II: $b_3 = 1.2 \ln(1+a_3)$ unit III: $c = 0.9\ b$		
$c \le 1$ $b_2 \le 5$		

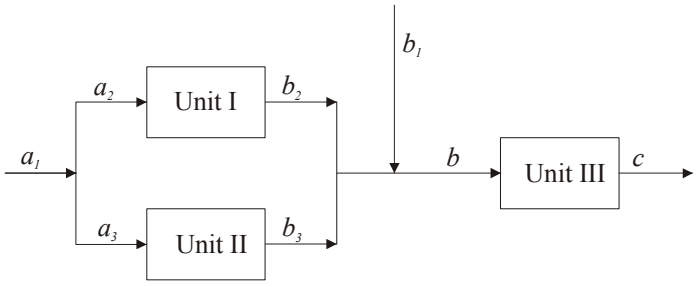


Figure 4.5. Example network

Fig. 4.5 is a one task – one equipment (OTOE) network, used for visualising the superstructure. The units are represented by nodes, and the streams are represented by edges. This figure does not represent a graph in mathematical sense because in graphs the edges must connect nodes, whereas here some edges (namely a_1 and b_1) originate from outside, and

one (namely *c*) is directed to outside of the process. For the sake of exact discussion, the R-graph representation of the same system is shown in Fig. 4.6.

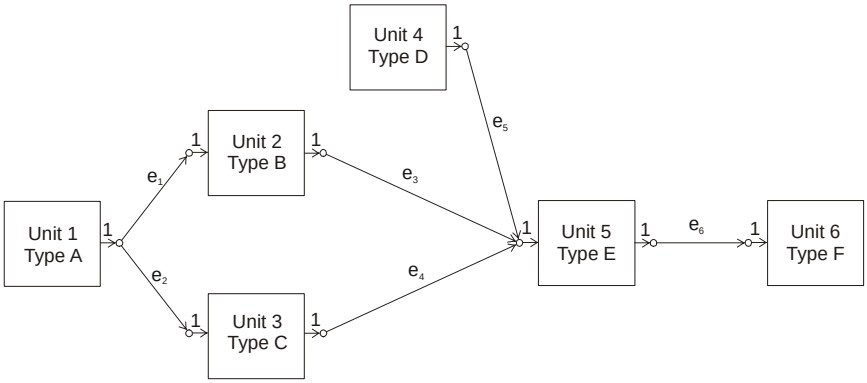


Figure 4.6. R-graph representation of Example 4.1. Letter 'e' stands for edges.

There are three additional units here. Unit 1 and Unit 4 are the sources of raw materials A and B; Unit 6 is the sink of product C. The source units do not have input ports, and the sink unit does not have output port.

In this example the considered structures are selected according to the following criteria:

1. Source stream(s) should exist. (That is, either stream a_1, or b_1, or both should be present.)
2. Product stream(s) should exist. (That is, stream *c* should be present.)
3. No other outlet stream than the product stream(s) may be present. (That is, only *c* may be an outlet stream here.)
4. Unit I and Unit II (in the graph representation: Unit 2 and Unit 3) should not exist simultaneously.

The first criterion states that the product is always made from the raw material(s), and cannot be made from nothing. The second criterion states that some product is to be produced. The third criterion expresses the special need of this example that no side-product is to be produced. The fourth criterion is applied because Unit I and Unit II (Unit 2 and Unit 3 in the graph representation) accomplish the same (or similar) transformation on the materials, even if they are units of different type, and application of two units of different types parallelly on the same task seems uneconomic. All these criteria are based on engineering considerations.

Before presenting the graph representation of the considered structures, please, check the network representation. The network shown in Fig. 4.7 is a subnetwork of the network shown

in Fig. 4.5. However, it does not satisfy the above criteria; thus, it represents a non-considered structure. The chance for such a problem to occur is significantly reduced by applying R-graph representation.

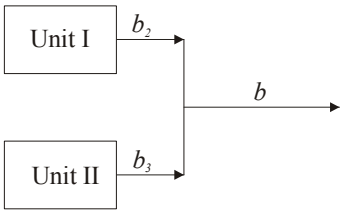


Figure 4.7. A subnetwork that represents a non-considered structure

Figs. 4.8a-g show all the subgraphs of the supergraph shown in Fig. 4.6. All these subgraphs satisfy the first three criteria. On the other hand, Graph 6 and Graph 7 (Figs. 4.8f-g) do not satisfy the fourth criterion. The first three criteria are automatically satisfied, and only the fourth criterion is to be checked when graphs are applied. Therefore, the use of R-graph representation is beneficial in automatic synthesis.

All the units of any supergraph are different in this simple example; there are no units of the same type. For this reason, there cannot exist isomorphic subgraphs; every substructure is represented by only one graph. Therefore, each graph representing a considered structure is also a considered graph.

How the subgraphs of this example are sorted according to Fig. 4.3 is shown in Fig. 4.9.

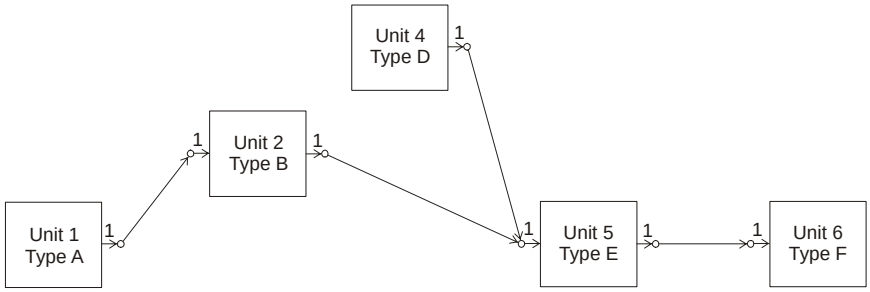


Figure 4.8a. Graph 1: A subgraph of the supergraph

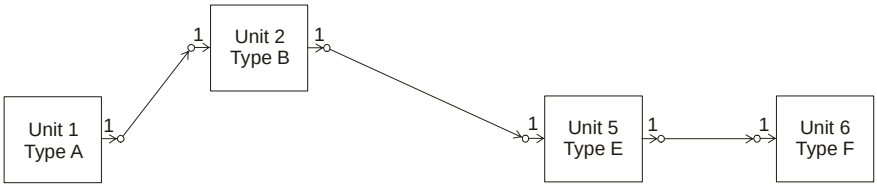


Figure 4.8b. Graph 2: A subgraph of the supergraph

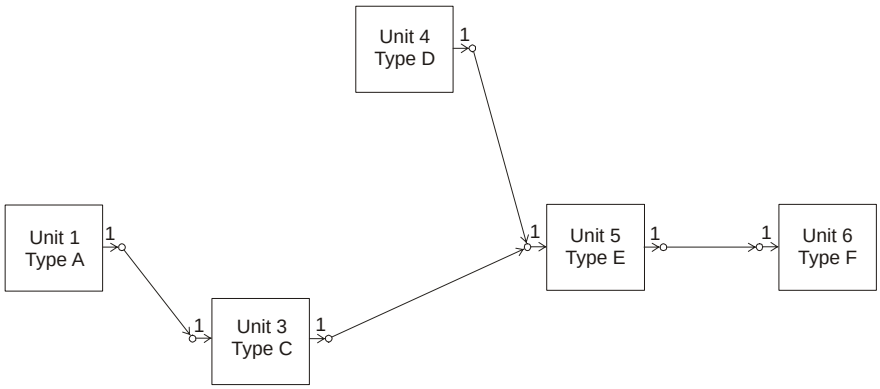


Figure 4.8c. Graph 3: A subgraph of the supergraph

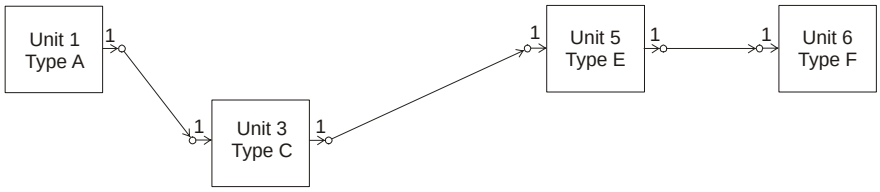


Figure 4.8d. Graph 4: A subgraph of the supergraph

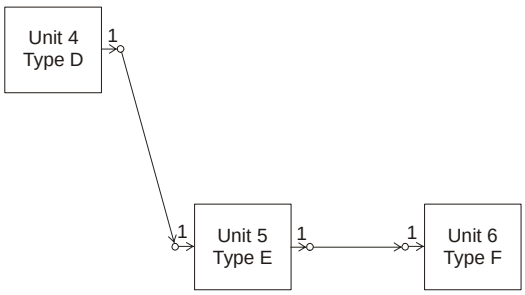


Figure 4.8e. Graph 5: A subgraph of the supergraph

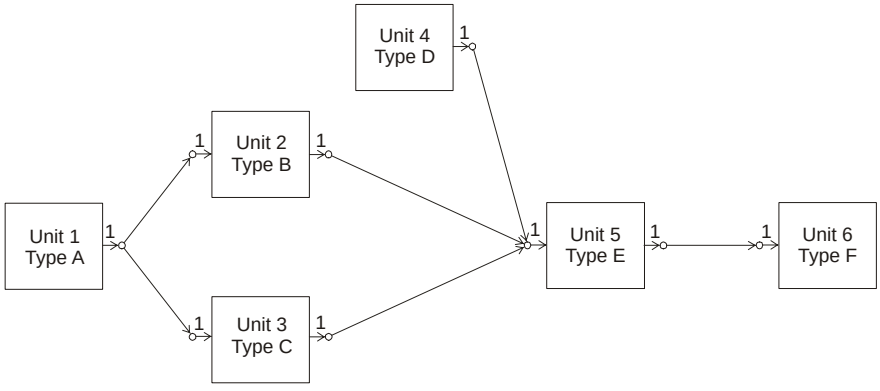


Figure 4.8f. Graph 6: A subgraph of the supergraph

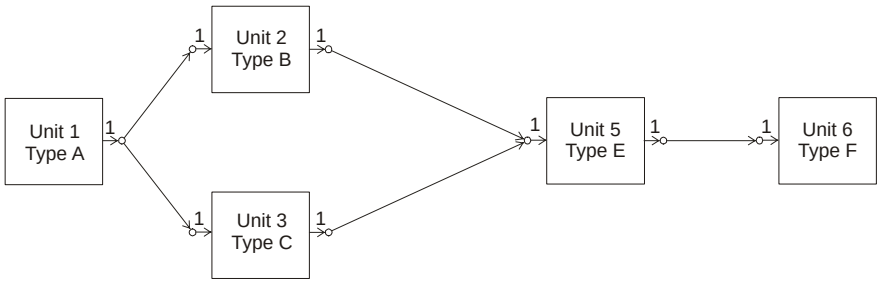


Figure 4.8g. Graph 7: A subgraph of the supergraph

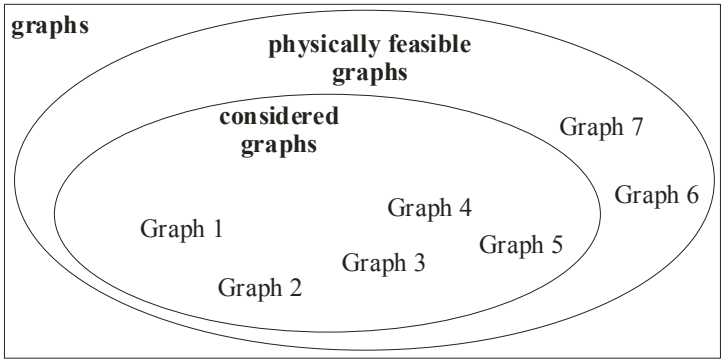


Figure 4.9. Grouping of the subgraphs

4.3. Redundancy in representing structures by MINLP problems

The above distinction between the set of considered structures and considered graphs is crucial in representing the structures by algebraic/logical models. Such a model contains variables; those variables correspond to uniquely labelled elements of the graph actually representing the superstructure. Therefore, the MINLP problem is not directly applied to the superstructure but to one of its particular representing graphs, the so-called supergraph. A solution of the MINLP problem assigns a subgraph of that particular representing supergraph. Graphs, representing structures, will be further represented by MINLP problems. As it was mentioned in the literature review, an MR, as an MINLP problem has the following form (Kocis and Grossmann, 1987; Grossmann, 1996):

$$\begin{aligned} \min \quad & OBJ = f(\mathbf{x},\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x},\mathbf{z}) \le \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} = \{\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \mathbf{x}^L \le \mathbf{x} \le \mathbf{x}^U\} \\ & \mathbf{z} \in \mathbf{Z} = \{\mathbf{z} \mid \mathbf{z} \in \{0,1\}^k,\ \mathbf{A}\mathbf{z} \le \mathbf{a}\} \end{aligned} \tag{4.1}$$

The feasible region of this MR is denoted by

$$\mathsf{FR}(\mathrm{MR}) = \left\{\mathbf{x}, \mathbf{z} \middle| \mathbf{x} \in \mathsf{X}, \mathbf{z} \in \mathsf{Z}, \mathbf{g}(\mathbf{x}, \mathbf{z}) \leq \mathbf{0}\right\} \tag{4.2}$$

i.e. the set of **x** and **z** that satisfy Eqs. 4.1.

The MR as usually applied to the synthesis problem is never unique. An MR as defined above is nothing to do with the synthesis problem itself unless an unambiguous mapping is given from the domain of the variable values of the MR to the domain of the value set of natural parameters of the synthesis problem, or at least to the graph representation of the superstructure.

The essential problems in mathematically defining the MR are representativeness and uniqueness. Whether an MR represents a superstructure and all its considered substructures should be determinable. An other crucial problem is comparison of two different MINLP representations to decide if some process flowsheets are represented by both of them. This seems a rather difficult task because infinite number of different MR-s can be generated just by simple variable transformations in such a way that the generated MR-s are equivalent.

In order to avoid ambiguousity in defining the variables of the MR, first a *Basic GDP Representation* (BGR) involving logical relations is applied, and then a so-called *Basic MINLP Representation* (BMR) is constructed with standard mapping between them. BGR can be constructed by applying a standard 'natural' representation of the process. Then it is easy to unambiguously transform the logical relations to algebraic ones. Equivalency and representativeness of MR-s in general form will then be analyzed by reducing them to their BMR or by comparing their feasible domains.

Once the definition of MR is properly given, ideality and redundancy can be analyzed. *Ideal MR* (IMR) and *Binarily Minimal MR* (BMMR) will be defined and compared as most important extremities.

4.4. Basic representations

The usual practice of developing an MINLP model of a synthesis problem is first intuitively constructing a superstructure of the given problem. The superstructure is usually represented by a network or, more conveniently, a graph (a supergraph). Then some state variables are also intuitively assigned to the elements (nodes and edges) of the graph. These variables may refer to thermodynamic states, operational parameters, construction parameters, and even to the existence of units and/or connections. These variables together with the evaluation

variables (parameters of the objective function, and the variable carrying the function value itself) constitute the set of variables of the MINLP model.

Constructing a proper MR is a difficult task demanding heavily on the engineer's abilities. Since logical relations are more human friendly than integer variables and relations, the logical formulation, e.g. the GDP representation, can be utilized as a first step in formulating the MR. Thus, first a so-called *Basic GDP Representation* (BGR) is constructed, then, in turn, a *Basic MINLP Representation* (BMR) automatically transformed from BGR. This BMR will serve as a reference for comparison to decide whether an arbitrary MINLP formulation represents the superstructure.

Representativeness and uniqueness are the two essential viewpoints in defining BGR and BMR. BMR is formed in a way as to represent only and all the subgraphs of a given supergraph. Numerical viewpoints can be taken into account later, in constructing the actual form of MR. The variables and the equations defined in BGR and BMR are not necessarily present in the final form of MR.

4.4.1. Basic GDP Representation

A General Disjunctive Programming (GDP) model is constructed in the spirit of Yeomans and Grossmann (1999a) but based on the R-graph representation given in Rév et al. (2005). This GDP model is called here *Basic GDP Representation* (BGR).

It is formulated in such a way that some formulas describing the behaviour of unit operations work on the variables belonging to particular units; some formulas work on stream properties, etc. Therefore the variables have to be grouped according to which unit operations and which streams they belong to.

A set of unit type variables and unit type equations are to be attributed to each unit type. When a particular unit (with particular labels, like “1” or “2” amongst the same type of units) is selected, it is attributed with particular unit equations and constraints (together called unit relations) according to its type, acting on the set of unit variables. These unit relations are of the same shape for identical unit types. That is, if there are two copies of the same unit type then two, formally equivalent, sets of relations are applied to different variables.

Basic GDP Representation (BGR) is then defined by the following formulation:

I. Sets

The sets include the frame set of units and unit types, their input and output ports as well. Any particular graph is a construction including the set $\mathbf{M}$ of actual units, the set of input and output ports, the types of these units, and the set $\mathbf{E}$ of graph edges e ($e \in \mathbf{E}$).

R-graph is a connected graph, and its subgraphs also have this property. In some cases one or more units of a graph occur in each of its subgraphs. For example, if a graph has exactly one sink unit, then this unit should occur in all its subgraphs, otherwise those subgraphs would not be R-graphs. Generally not just the sink or source units, but a unit of any type may have this property in a particular supergraph. These units are called *permanent units* of the supergraph, while all the other units are called *conditional units*.

Accordingly, the set $\mathbf{M}$ of units is partitioned in BGR as $\mathbf{M} = \mathbf{M}_{\text{perm}} \cup \mathbf{M}_{\text{cond}}$ where $\mathbf{M}_{\text{perm}}$ is the set of units that occur in all the subgraphs, and $\mathbf{M}_{\text{cond}}$ is the set of units that does not occur in all the subgraphs of the supergraph.

II. Variables

Each unit $m \in \mathbf{M}$ is attributed with the following variables:

Numerical variables:

$\mathbf{e}_{m,r}^{in}$	array of inlet extensive variables, r=1, 2, …, α_t
$\mathbf{e}_{m,r}^{out}$	array of outlet extensive variables, r=1, 2, …, β_t
$\mathbf{i}_{m,r}^{in}$	array of inlet intensive variables, r=1, 2, …, α_t
$\mathbf{i}_{m,r}^{out}$	array of outlet intensive variables, r=1, 2, …, β_t
$\mathbf{d}_m$	array of design and control variables
$\mathbf{o}_m$	array of operation variables
c_m^{fix}	fix costs due to unit m
c_m^{var}	variable costs due to unit m

where

t	type of unit m
α_t	number of input ports of unit type t
β_t	number of output ports of unit type t

The variables in the arrays $\mathbf{e}_{m,r}^{in}$ and $\mathbf{i}_{m,r}^{in}$ will be lumped together and denoted by $\mathbf{x}_{m}^{in}$, and the lumped version of $\mathbf{e}_{m,r}^{out}$ and $\mathbf{i}_{m,r}^{out}$ is denoted by $\mathbf{x}_{m}^{out}$.

For each conditional unit a logical variable is also defined:

Z_m the existence of conditional unit *m*

Variables attributed to the edges of the graph, describing what fraction of the stream produced at the output port of a unit where that edge starts is directed to the input port of an other unit, where the edge points to:

$$0 \le \varphi_e \le 1 \tag{4.3}$$

III. Feasibility constraints

Unit relations:

$$\left.\begin{array}{l} \mathbf{x}_m^{in} \ge \mathbf{0};\ \mathbf{x}_m^{out} \ge \mathbf{0} \\ \mathbf{x}_m^{in} \ne \mathbf{0};\ \mathbf{x}_m^{out} \ne \mathbf{0} \\ \mathbf{o}_m \ge \mathbf{0};\ \ \mathbf{d}_m \ge \mathbf{0} \\ \mathbf{P}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \le \mathbf{0} \\ c_m^{fix} = \mathrm{Pfix}_t(\mathbf{d}_m) \\ c_m^{var} = \mathrm{Pvar}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \end{array}\right] \quad \text{for all } m \in \mathsf{M}_{\mathrm{perm}} \tag{4.4a}$$

$$\left[\begin{array}{l} Z_m \\ \wedge \\ \mathbf{x}_m^{in} \ge \mathbf{0};\ \mathbf{x}_m^{out} \ge \mathbf{0} \\ \mathbf{x}_m^{in} \ne \mathbf{0};\ \mathbf{x}_m^{out} \ne \mathbf{0} \\ \mathbf{o}_m \ge \mathbf{0};\ \ \mathbf{d}_m \ge \mathbf{0} \\ \mathbf{P}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \le \mathbf{0} \\ c_m^{fix} = \mathrm{Pfix}_t(\mathbf{d}_m) \\ c_m^{var} = \mathrm{Pvar}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \end{array}\right] \vee \left[\begin{array}{c} \neg Z_m \\ \wedge \\ \mathbf{x}_m^{in} = \mathbf{0};\ \mathbf{x}_m^{out} = \mathbf{0} \\ \mathbf{o}_m = \mathbf{0};\ \mathbf{d}_m = \mathbf{0} \\ c_m^{fix} = 0 \\ c_m^{var} = 0 \end{array}\right] \quad \text{for all } m \in \mathsf{M}_{\mathrm{cond}} \tag{4.4b}$$

Equations for input ports:

$$\mathbf{e}_{m,r}^{in} = \sum_{e \in \mathsf{E}[m,in,r]} \varphi_e \mathbf{e}_{n,h}^{out} \qquad \text{for all } <m,in,r> \tag{4.5a}$$

$$\mathbf{i}_{m,r}^{in} = \mathrm{f}^{i}(\mathbf{x}_{n,h}^{out}) \qquad \text{for all } <m,in,r> \tag{4.5b}$$

where $\mathsf{E}[m,in,r]$ is the set of all the edges ending at input port *r* of unit *m*; these edges are started at output ports <*n,out,h*>, where <*n,out,h*> is the output port *h* of unit *n* that is

connected to input port *r* of unit *m* by an edge *e*. The array $\mathbf{x}_{n,h}^{out}$ includes all the extensive and intensive variables of all the ports <*n,out,h*> connected to port <*m,in,r*> by an edge.

Equations for output ports:

$$\sum_{e \in \mathsf{E}[m,out,k]} \varphi_e = 1 \qquad \text{for all } <m,out,k> \tag{4.6}$$

IV. Objective function

$$\min_{\mathbf{x},\mathbf{z}} \sum_{m \in \mathsf{M}} \left(c_m^{fix} + c_m^{var} \right) \tag{4.7}$$

The above defined sets, variables, and Eqs. 4.3-4.7 together define the Basic GDP Representation (BGR).

Eq. 4.6 expresses the requirement that the sum of fractions is unity. Naturally, these fractions are non-negative, and limited by 1. This is expressed by Eq. 4.3.

Eqs. 4.4 express the feasibility constraints and cost functions attributed to the unit operations. For the sake of simplicity, negative variables are excluded. Any negative number can be expressed as a difference of two non-negative numbers. $\mathbf{P}_t$ defines the unit operation together with its equipment (construction). Generally, subequation system $\mathbf{P}_t$ includes components of both equality and inequality form. For example, a material balance around a unit is an equality, whereas a subequation expressing that a variable is greater than some minimum, as a complicated function of the other variables, is an inequality. Any equality can be expressed as a pair of two inequalities. For the sake of simplicity in notation and the proofs, "smaller than or equal to" relation is used here for $\mathbf{P}_t$ in Eqs. 4.4. On the other hand, all the results remain valid if equality may occur in components of $\mathbf{P}_t$; and application of equality, if possible, is simpler in practice.

The objective function (Eq. 4.7) is a sum of objective parts attributed to each unit. These parts are computed by Pfix and Pvar. Pfix is the part of fix costs, depending on the design and control variables. Pvar expresses the variable costs. Eq. 4.4a is applied to the permanent units; Eq. 4.4b is applied to the conditional units. Variable Z_m in Eq. 4.4b expresses the existence or non-existence of the unit *m*. If the unit exists, the variables should satisfy the same equations that occur in Eq. 4.4a ($\mathbf{P}_t$, Pfix, Pvar). When the unit does not exist, the cost increments must be zero, and all the other variables are also set to zero.

In Eq. 4.4b logical relations $\wedge$ ('and') and $\vee$ ('or') and $\neg$ ('not') take place; this is a logical truth function. Each conditional unit is defined in the so-called disjunctive normal form; that is why this model is called GDP (disjunctive programming).

In the model the output ports behave as stream splitters (Eq. 4.6); the input ports behave as stream unifiers (Eqs. 4.5). In the unifiers the extensive variables are added together (Eq. 4.5a), while the intensive variables of the unified stream are more complicated functions of all the stream properties (Eq. 4.5b).

A significant difference between the BGR and the GDP model of Yeomans and Grossmann (1999a) is that in BGR not any additional pure logical relations is provided between the logical variables (see Eqs. (4) and (18) in Yeomans and Grossmann, 1999a) that would look in this case as:

$$\Omega(\mathbf{Z})=\text{True} \tag{4.8}$$

There are two reasons for this change. Eq. 4.8 would be used for expressing possible substructures, and, based on engineering considerations, for excluding some not considered substructures from the set of structures. In our case, however, the supergraph is an R-graph, therefore no additional logical relations are needed to define the substructures. Instead, Eqs. 4.5-4.6 express the splitter and unifier properties. On the other hand, here a basic representation is automatically constructed, and it does not include engineering considerations, that cannot be graphically represented. Such considerations will be included in a later phase of the modelling and design procedure.

Another significant difference is that the Pvar and Pfix functions are not inserted directly into the objective function, but refer to them through the cost variables c_m^{var} and c_m^{fix}. Although the significance of this choice is not evident in the first sight, it will be transparent when the MINLP representation is formed. The essential consequence of this small change is that the binary variables in this model are not present in the objective function.

A third difference is that the assignment of permanent units is used by Yeomans and Grossmann for arbitrarily constraining the set of feasible structures. In BGR only those units are assigned as being permanent that really are present in all the subgraphs. This is an unambiguous assignment.

4.4.2. Example 4.1 – Basic GDP representation

Here the GDP representation of Example 4.1, given in Table 4.1 and Figs. 4.4-4.9, is constructed.

The continuous variables represent the inlet and outlet flow rates and the costs. There are not intensive or other extensive variables of streams. Neither design nor operation variables of units are applied.

Continuous variables.

x_m^{in} inlet flow rate (ton/h)

x_m^{out} outlet flow rate (ton/h)

c_m^{fix} fix costs due to the existence of unit m

c_m^{var} variable costs belonging to unit m

The permanent units are Unit 5 and Unit 6 because they belong to the only possible product. All the other units are conditional because either one of Unit 1 or Unit 4 may play the role of a single feed, and either Unit 2 or Unit 3 may be present in the solution without the other one. A Z_m logical variable is also defined to each conditional unit. Finally, a continuous variable φ_e is assigned to each edge. There are altogether 27 continuous real variables (x_i, i=1, 2, … 27), including all the x_m^{in}, x_m^{out}, c_m^{fix}, c_m^{var}, and φ_e variables, and 4 logical variables (Z_m, m=1, 2, 3, 4). These variables can be lumped in an array **x** of the continuous variables, and an array **Z** of the logical variables. Once these variables are assigned, their bounds can be defined. The bounds assign a range $\mathbf{X}$ of the continuous array, and a range $\overline{\mathbf{Z}}$ of the logical variables, as in Eq. 4.9.

$$\mathbf{X}=\left\{\begin{bmatrix} x_1^{out} \\ x_2^{in} \\ x_2^{out} \\ x_3^{in} \\ x_3^{out} \\ x_4^{out} \\ x_5^{in} \\ x_5^{out} \\ x_6^{in} \\ c_1^{fix} \\ c_2^{fix} \\ c_3^{fix} \\ c_4^{fix} \\ c_5^{fix} \\ c_6^{fix} \\ c_1^{var} \\ c_2^{var} \\ c_3^{var} \\ c_4^{var} \\ c_5^{var} \\ c_6^{var} \\ \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_5 \\ \varphi_6 \end{bmatrix} \middle| \mathbf{x}\in\mathbb{R}^{27}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 3.5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -13 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \le \begin{bmatrix} x_1^{out} \\ x_2^{in} \\ x_2^{out} \\ x_3^{in} \\ x_3^{out} \\ x_4^{out} \\ x_5^{in} \\ x_5^{out} \\ x_6^{in} \\ c_1^{fix} \\ c_2^{fix} \\ c_3^{fix} \\ c_4^{fix} \\ c_5^{fix} \\ c_6^{fix} \\ c_1^{var} \\ c_2^{var} \\ c_3^{var} \\ c_4^{var} \\ c_5^{var} \\ c_6^{var} \\ \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_5 \\ \varphi_6 \end{bmatrix} \le \begin{bmatrix} 3.57 \\ 2.04 \\ 5 \\ 1.53 \\ 1.11 \\ 1.11 \\ 1.11 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1.5 \\ 0 \\ 3.5 \\ 0 \\ 6.42 \\ 5 \\ 2.31 \\ 7.78 \\ 2 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\} \cup \overline{\mathbf{Z}}=\left\{\begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \end{bmatrix} \middle| Z_m \in \{\text{true}, \text{false}\}\right\} \tag{4.9}$$

Now, the relations can be given as follow.

<u>Unit relations of permanent units:</u>

Unit 5

$$\begin{bmatrix} x_5^{in} > 0; \quad x_5^{out} > 0 \\ x_5^{out} - 0.9\, x_5^{in} \le 0 \\ -x_5^{out} + 0.9\, x_5^{in} \le 0 \\ c_5^{fix} = 3.5 \\ c_5^{var} = 2\, x_5^{out} \end{bmatrix} \tag{4.10a}$$

Unit 6

$$\begin{bmatrix} x_6^{in} > 0 \\ c_6^{fix} = 0 \\ c_6^{var} = -13\, x_6^{in} \end{bmatrix} \tag{4.10b}$$

Unit relations of conditional units:

Unit 1

$$\begin{bmatrix} Z_1 \\ \wedge \\ x_1^{out} > 0 \\ c_1^{fix} = 0 \\ c_1^{var} = 1.8\, x_1^{out} \end{bmatrix} \vee \begin{bmatrix} \neg Z_1 \\ \wedge \\ x_1^{out} = 0 \\ c_1^{fix} = 0 \\ c_1^{var} = 0 \end{bmatrix} \tag{4.10c}$$

Unit 2

$$\begin{bmatrix} Z_2 \\ \wedge \\ x_2^{in} > 0; \quad x_2^{out} > 0 \\ x_2^{out} - \ln\left(x_2^{in} + 1\right) \le 0 \\ -x_2^{out} + \ln\left(x_2^{in} + 1\right) \le 0 \\ c_2^{fix} = 1 \\ c_2^{var} = x_2^{out} \end{bmatrix} \vee \begin{bmatrix} \neg Z_2 \\ \wedge \\ x_2^{in} = 0; \quad x_2^{out} = 0 \\ c_2^{fix} = 0 \\ c_2^{var} = 0 \end{bmatrix} \tag{4.10d}$$

Unit 3

$$\begin{bmatrix} Z_3 \\ \wedge \\ x_3^{in} > 0; \quad x_3^{out} > 0 \\ x_3^{out} - 1.2\ln\left(x_3^{in} + 1\right) \le 0 \\ -x_3^{out} + 1.2\ln\left(x_3^{in} + 1\right) \le 0 \\ c_3^{fix} = 1.5 \\ c_3^{var} = 1.2\, x_3^{out} \end{bmatrix} \vee \begin{bmatrix} \neg Z_3 \\ \wedge \\ x_3^{in} = 0; \quad x_3^{out} = 0 \\ c_3^{fix} = 0 \\ c_3^{var} = 0 \end{bmatrix} \tag{4.10e}$$

Unit 4

$$\begin{bmatrix} Z_4 \\ \wedge \\ x_4^{out} > 0 \\ c_4^{fix} = 0 \\ c_4^{var} = 7\, x_4^{out} \end{bmatrix} \vee \begin{bmatrix} \neg Z_4 \\ \wedge \\ x_4^{out} = 0 \\ c_4^{fix} = 0 \\ c_4^{var} = 0 \end{bmatrix} \tag{4.10f}$$

Source and sink units do not have operation equations. The operation equations of the other units are formed as inequalities. (These relations can be written in equation form in the MINLP representation.)

Equations for input ports:

$$\begin{aligned} x_2^{in} &= \varphi_1 \cdot x_1^{out} \\ x_3^{in} &= \varphi_2 \cdot x_1^{out} \\ x_5^{in} &= \varphi_3 \cdot x_2^{out} + \varphi_4 \cdot x_3^{out} + \varphi_5 \cdot x_4^{out} \\ x_6^{in} &= \varphi_6 \cdot x_6^{out} \end{aligned} \tag{4.11}$$

Equations for output ports:

$$\begin{aligned} &\varphi_1 + \varphi_2 = 1 \\ &\varphi_3 = 1 \\ &\varphi_4 = 1 \\ &\varphi_5 = 1 \\ &\varphi_6 = 1 \end{aligned} \tag{4.12}$$

Those φ_e variables which have value 1 may be dropped in an MINLP representation different from BGR and BMR.

Objective function:

$$\min_{\mathbf{x},\mathbf{z}} \sum_{m=1}^{6} \left(c_m^{fix} + c_m^{var} \right) \tag{4.13}$$

4.4.3. Basic MINLP Representation

BGR can be solved using appropriate algorithms utilizing the GDP form; but representing the synthesis problem by MINLP model is yet more common and well-spread in the chemical engineering communities. However, well formulating a synthesis problem as MINLP is difficult, whereas formulating it as a GDP is most convenient. Respectably, it is worth *automatically transforming* the conveniently formulated BGR of the synthesis problem to its MINLP formulation.

BMR is automatically formed from BGR by transforming the disjunctions of Eq. 4.4b into algebraic form while substituting binary variables *z* in the place of the logical variable *Z.* As it was mentioned in the literature review, widespread transformations are the so-called Big M,

Multi M and Convex hull methods (Raman and Grossmann, 1991; Grossmann and Türkay, 1996; Szitkai et al., 2002; Vecchietti et al., 2003). Any of these methods may be used for defining BMR; all are equally proper methodologies. For example, the Big M transformation of Eq. 4.4b leads to the following subsystem of equations:

$$
\begin{aligned}
&\mathbf{x}_m^{in} \le \mathbf{U}_m^{in} z_m \\
&\boldsymbol{\varepsilon} - \mathbf{x}_m^{in} \le \mathbf{U}_m^{in}(1 - z_m) \\
&\mathbf{x}_m^{out} \le \mathbf{U}_m^{out} z_m \\
&\boldsymbol{\varepsilon} - \mathbf{x}_m^{out} \le \mathbf{U}_m^{out}(1 - z_m) \\
&\mathbf{0} \le \mathbf{o}_m \le \mathbf{U}_m^{o} z_m \\
&\mathbf{0} \le \mathbf{d}_m \le \mathbf{U}_m^{d} z_m \\
&0 \le c_m^{fix} \le U_m^{fix} z_m \\
&L_m^{var} z_m \le c_m^{var} \le U_m^{var} z_m \\
&\mathbf{L}_m^{P} z_m \le \mathbf{P}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \le \mathbf{U}_m^{P}(1 - z_m) \\
&L_m^{Pfix}(1 - z_m) \le c_m^{fix} - \mathrm{Pfix}_t(\mathbf{d}_m) \le U_m^{Pfix}(1 - z_m) \\
&L_m^{Pvar}(1 - z_m) \le c_m^{var} - \mathrm{Pvar}_t(\mathbf{x}_m^{in}, \mathbf{x}_m^{out}, \mathbf{d}_m, \mathbf{o}_m) \le U_m^{Pvar}(1 - z_m)
\end{aligned}
\qquad (4.14–4.24)
$$

where L and U are the lower and upper bounds, respectively, and $\boldsymbol{\varepsilon}$ is a non-zero vector of small length, introduced in order to exclude zero vector from the domain.

4.4.4. Example 4.1 – Basic MINLP Representation

Once the GDP unit relations of the conditional units are given (Eqs. 4.10c-f), the *Basic MINLP Representation* can be automatically generated. First, the logical variables Z are transformed into binary *z*ones; thus, the range $\mathbf{Z}$ of the binary variables becomes:

$$
\mathbf{Z} = \left\{ \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} \middle| \mathbf{z} \in \{0,1\}^4 \right\} \qquad (4.25)
$$

Then, the equations can also be transformed, accordingly. The transformed equations are listed below:

$$
\begin{aligned}
&-2.04(1 - z_2) + \varepsilon \le x_2^{in} \le 2.04 z_2 \\
&-1.53(1 - z_3) + \varepsilon \le x_3^{in} \le 1.53 z_3
\end{aligned}
\qquad (4.26)
$$

$$\begin{aligned}
&-3.57(1-z_1)+\varepsilon \le x_1^{out} \le 3.57 z_1\\
&-1.11(1-z_2)+\varepsilon \le x_2^{out} \le 1.11 z_2\\
&-1.11(1-z_3)+\varepsilon \le x_3^{out} \le 1.11 z_3\\
&-1.11(1-z_4)+\varepsilon \le x_4^{out} \le 1.11 z_4
\end{aligned} \tag{4.27}$$

$$\begin{aligned}
&0 \le c_1^{fix} \le 0 z_1\\
&0 \le c_2^{fix} \le 1 z_2\\
&0 \le c_3^{fix} \le 1.5 z_3\\
&0 \le c_4^{fix} \le 0 z_4
\end{aligned} \tag{4.28}$$

$$\begin{aligned}
&0 z_1 \le c_1^{var} \le 6.42 z_1\\
&0 z_2 \le c_2^{var} \le 5 z_2\\
&0 z_3 \le c_3^{var} \le 2.31 z_3\\
&0 z_4 \le c_4^{var} \le 7.78 z_4
\end{aligned} \tag{4.29}$$

$$\begin{aligned}
&-1.11 z_2 \le x_2^{out} - \ln\left(x_2^{in}+1\right) \le 1.11(1-z_2)\\
&-1.11 z_2 \le -x_2^{out} + \ln\left(x_2^{in}+1\right) \le 1.11(1-z_2)\\
&-1.11 z_3 \le x_3^{out} - 1.2\ln\left(x_3^{in}+1\right) \le 1.11(1-z_3)\\
&-1.11 z_3 \le -x_3^{out} + 1.2\ln\left(x_3^{in}+1\right) \le 1.11(1-z_3)
\end{aligned} \tag{4.30}$$

$$\begin{aligned}
&0(1-z_1) \le c_1^{fix} - 0 \le 0(1-z_1)\\
&-1(1-z_2) \le c_2^{fix} - 1 \le 0(1-z_2)\\
&-1.5(1-z_3) \le c_3^{fix} - 1.5 \le 0(1-z_3)\\
&0(1-z_4) \le c_4^{fix} - 0 \le 0(1-z_4)
\end{aligned} \tag{4.31}$$

$$\begin{aligned}
&-6.42(1-z_1) \le c_1^{var} - 1.8 x_1^{out} \le 6.42(1-z_1)\\
&-5(1-z_2) \le c_2^{var} - x_2^{out} \le 5(1-z_2)\\
&-2.31(1-z_3) \le c_3^{var} - 1.2 x_3^{out} \le 2.31(1-z_3)\\
&-7.78(1-z_4) \le c_4^{var} - 7 x_4^{out} \le 7.78(1-z_4)
\end{aligned} \tag{4.32}$$

Some of these equations can be written in simpler form. For example, a simpler form of the equations concerned to c_4^{fix} (from Eqs. 4.28 and 4.31) is the following:

$$c_4^{fix} = 0 \tag{4.33}$$

Instead of the four inequalities in Eq. 4.32, two inequalities are enough:

$$\begin{aligned}
&-1.11 z_2 \le x_2^{out} - \ln\left(x_2^{in}+1\right) \le 1.11(1-z_2)\\
&-1.11 z_3 \le x_3^{out} - 1.2\ln\left(x_2^{in}+1\right) \le 1.11(1-z_3)
\end{aligned} \tag{4.34}$$

These simpler forms (Eqs. 4.33-4.34) are, however, not used here in order to ensure the unambiguous form of the BMR.
The non-transformed variables and the non-transformed equations are unchanged; therefore they are not shown here.

4.5. MINLP representation

For avoiding multiplicity and redundancy in the MINLP problem representation, one has to be able to determine if a given MR represents a supergraph (or a superstructure) or not. The same question emerges if two representations have to be compared according to their feasible regions. Without special care, one cannot be sure if the MR really represents all the considered structures. I have not found in the literature such a definition of MR that could be applied to solve this problem.
The main difficulty here lies in the fact that the number of variables, and even their names and characteristics are subject to arbitrary variations.
Here the definition of the MR is suggested through a fixed form of BMR. There is a double merit of this definition. First, in this way the question of representation can be solved, as shown below. Second, such a BMR can automatically be generated and can serve as a reference representation.
MR should be defined in such a way that each of its solutions unambiguously assigns a graph state and thus a flowsheet in the same way as BMR does, but it may contain arbitrary superfluous information. This may include superfluous information on structures not considered, or it may also include redundant information on the considered structures.
Here MR is defined in two different, but related, ways. The first, general, definition deals with the ability of an MINLP problem formulation to represent the considered structures of the synthesis task. This general definition includes the existence of a bijection π between FR(BMR) and a subset B of FR(MR). This definition can be applied to check if an MINLP problem formulation can anyhow represent the considered structures of a synthesis task.
The general definition of MR is the following:
An MR (MINLP problem Representation) represents a graph if the following two conditions are satisfied:

1. Such a subregion $\mathsf{B} \subseteq \mathsf{FR}$(MR) exists that a bijection π: $\mathsf{B} \Leftrightarrow \mathsf{FR}$(BMR) can be given where FR(BMR) is the feasible region of the basic representation of the graph, and FR(MR) is the feasible region of the actual MR in question.

2. For each solution (x, z) of MR that lies in B (that is $(x, z) \in \mathsf{B}$) $OBJ_{MR}(x, z)$ is equal to $OBJ_{BMR}(\pi(x, z))$, where OBJ_{MR} and OBJ_{BMR} denote the value of the objective functions of MR and BMR, respectively.

This general definition of MR is explained in Fig. 4.10. The constraining conditions, that enforce the solution to represent an R-graph, assign the feasible region FR(BMR). Points of FR(BMR) describe the subgraphs (but not necessarily just the considered graphs). B is a subset of the feasible region of MR, that is, it is a subset of FR(MR). If a part of B was not subset of FR(MR) then some of the feasible solutions of BMR, and thus some of the subgraphs, would not be represented by MR. As B is a subset of FR(MR), the feasible solutions of MR include all the solutions that are mapped to the states of the subgraphs. The connections between the representations and sets are shown in Fig. 4.11.

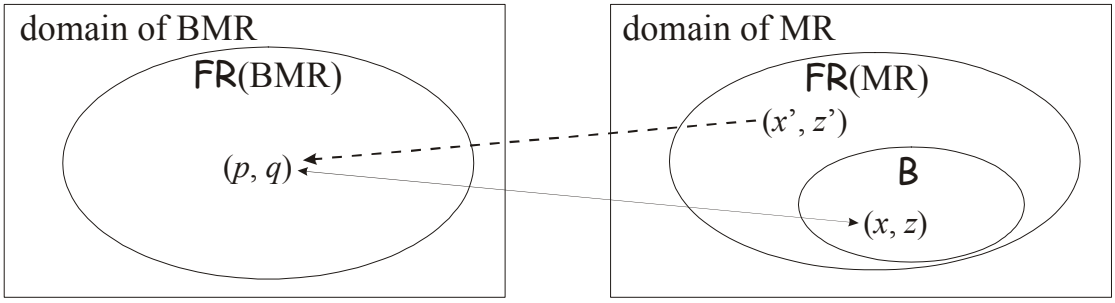


Figure 4.10. The mapping π, and the relation between FR(BMR), and FR(MR)

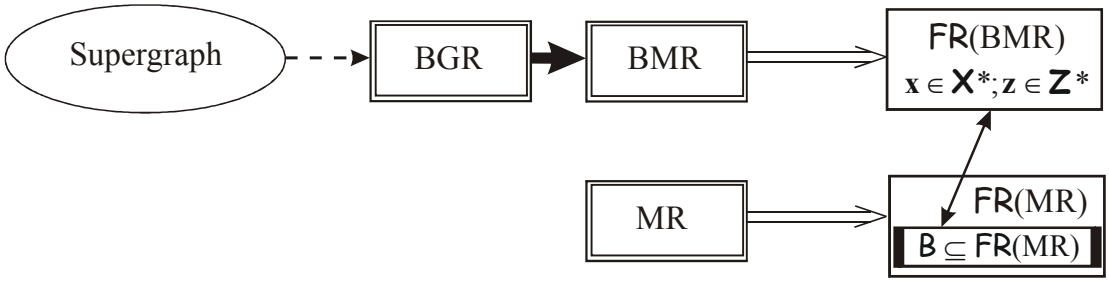


Figure 4.11. Connections between sets and representations

Here the reader has to be reminded to what was written in the paragraph below Eq. 4.8. The superstructure is represented by a supergraph, and no structural constraints additional to what

is applied in BGR are applied in BMR. Therefore, $\mathbf{FR}$(BMR) includes all the subgraphs of the supergraph, not just the considered graphs. Thus, all the subgraphs of the supergraph are represented by $\mathbf{FR}$(BMR) and by $\mathbf{B}\subseteq\mathbf{FR}$(MR). It then follows that if an R-graph is represented by an MR then all the subgraphs of that R-graph are also represented by that MR.

According to the definition of isomorphy, bijection can always be given between two isomorphic graphs. It then follows that if a graph is represented by an MR then all its isomorphic graphs are also represented by that MR.

$\mathbf{FR}$(MR) may contain feasible solutions outside of $\mathbf{B}$, and these solutions may correspond to graphs and structures not included in $\mathbf{FR}$(BMR). Such a correspondence can be expressed by some mapping from $\mathbf{FR}$(MR) to the set of some flowsheets, or their representing graphs, including all those represented by $\mathbf{FR}$(BMR), and others not represented by it. That is, an arbitrary representation of the synthesis task can be wider than the BMR, but cannot be narrower. In this way a representation can carry any additional information, but it also carries all the information necessary for describing the subgraphs of the supergraph.

The second, restricted, definition of MR includes a particularly assigned surjection ψ from $\mathbf{FR}$(MR) to $\mathbf{FR}$(BMR). Such a mapping ψ can be assigned only if the above bijection π exists. Once such a surjection ψ is given, a bijection π with the property $\pi\subseteq\psi$ always exists. Thus, this representation can be defined as a restriction of the general MR to the application of a particularly selected surjection ψ.

MR may contain a feasible solution (x', z'), outside of B, that is also mapped to (p, q) according to the surjection ψ, as is also shown in Fig. 4.10. In this case the MR with the particular selection of ψ is redundant, because two feasible solutions are mapped to the same flowsheet. Such a redundancy is not excluded, and avoiding this kind of redundancy is not always preferable.

This restricted definition of MR is beneficial in defining ideal MINLP representation. Which definition is applied must be clear from the textual environment.

We cannot emphasize with great enough weight that the actual MINLP problem formulation of MR is not in any way bound to the variables and equations of BMR. The only restriction is the existence of a bijection between a subset $\mathbf{B}\subset\mathbf{FR}$(MR) and $\mathbf{FR}$(BMR) in such a way as to provide with the same objective value. The engineer has the freedom to apply a formulation best fitting to convenience and numerical efficiency.

4.5.1. Example 4.1 – an MINLP representation

Kocis and Grossmann (1987) presented an MINLP representation to a planning problem identical to our example. This MR is also presented in Appendix. Here I demonstrate that the MR of Kocis and Grossmann (1987) represents the same superstructure as our BMR does for Example 4.1 presented in section 4.2 and also dealt with in subsection 4.4.2. For this aim, a bijective mapping from subregion B of FR(MR) to FR(BMR) has to be given.

It is shown the construction of this mapping. First, consider the source of raw material B that is represented by continuous variable b_1 in the network of Kocis and Grossmann (Fig. 4.5), and by a pair of continuous variable x_4^{out} and the binary variable z_4 in our Basic MINLP Representation. Variable x_4^{out} describes the flow rate of the material flow if that raw material is applied. Whether raw material B is applied is described by the binary variable z_4; this formally corresponds to the existence of Unit 4 (source unit). Here a mapping is given between the values of the pair (z_4, x_4^{out}) falling in the feasible region of BMR and the values of b_1 falling in the B subset of the feasible region of MR. (If MR represents all the structures represented by BMR then all the feasible values of (z_4, x_4^{out}) should have a picture value b_1, and each such picture value should have an *unambiguous* (z_4, x_4^{out}) ancestor value. On the other hand, the b_1 values in FR(MR) outside of B do not have ancestor according to this mapping.)

In the MR, b_1 is a non-negative variable without any upper bound; thus, b_1 can take any non-negative value. But the set of feasible values of b_1 is narrower. This feasible set can be determined as follows. There is an upper bound for the product c:

$$c \le c^{UP} = 1 \tag{4.35}$$

The operation equation of Unit III is:

$$c - 0.9b = 0 \tag{4.36}$$

From here, the maximal value that b can take is

$$b^{max} = c^{UP} / 0.9 = 1.11. \tag{4.37}$$

The material balance of mixing b_i flows is

$$b_1 + b_2 + b_3 - b = 0 \tag{4.38}$$

The values of either b_2 or b_3 can be arbitrary near zero. Therefore, the maximal value that b_1 can take is almost the maximal value of b. For practical purposes, we take the limit:

$$b_1^{max} = b^{max} = 1.11 \tag{4.39}$$

The feasible set of b_1 is, therefore, the closed interval [0, 1.11] (see **Fig. 4.12**).

The lower and upper bounds of the variables in the graph representation were determined in Eqs. 4.9 and 4.25. For z_4 and x_4^{out} variables these are:

$$\begin{aligned} z_4 &\in \{0,1\} \\ x_4^{out} &\in [0,1.11] \end{aligned} \tag{4.40}$$

However, there are additional constraints for these variables in BMR in order to exclude some extreme and prohibited cases:

$$x_4^{out} \le 1.11 z_4 \tag{4.41a}$$

$$-x_4^{out} \le 1.11(1-z_4) - \varepsilon \tag{4.42b}$$

Eqs. 4.40-4.41 describe the corresponding subset, belonging to these variables in BMR, of the feasible set of FR(BMR).

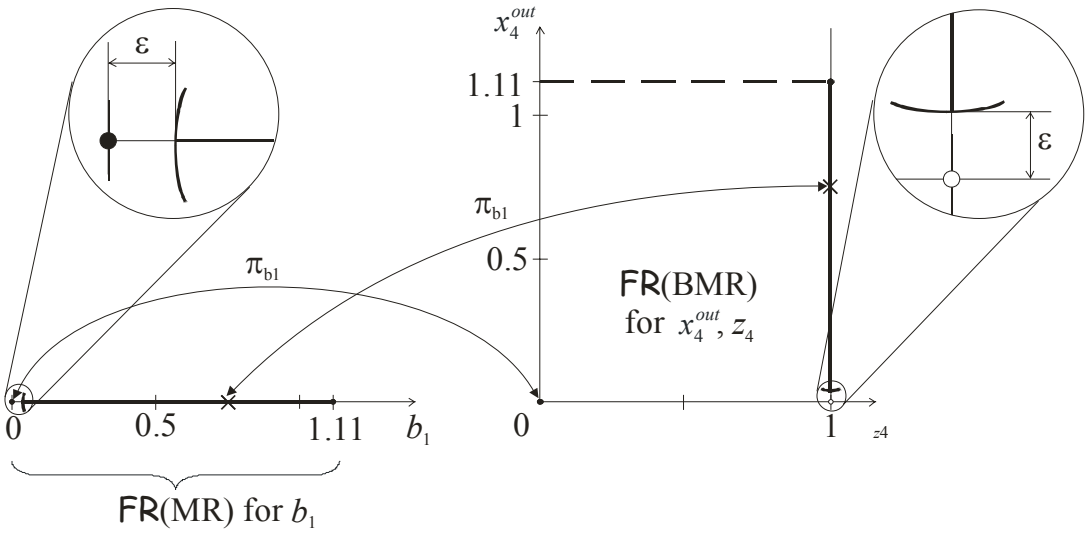


Figure 4.12. Bijection π applied to the feasible values of variable b_1

A bijective mapping π_{b1} between the feasible sets of b_1 and (z_4, x_4^{out}) can be defined as given by the analytical definition Eq. 4.42 and shown in Fig. 4.12:

$$\pi_{b1}: \begin{cases} \text{if } b_1 = 0 & \Rightarrow x_4^{out} = b_1 \text{ and } z_4 = 0 \\ \text{if } \varepsilon \le b_1 \le 1.11 & \Rightarrow x_4^{out} = b_1 \text{ and } z_4 = 1 \end{cases} \tag{4.42}$$

In bijection π_{b1}, variable b_1 can take values from region B_{b1}={0; [ε, 1.11]}, that is the union of the point of the exact zero and of the closed section between ε and 1.11. This set B_{b1} is the subregion of the feasible set of b_1 in FR(MR). For all the other variables in MR of Kocis and Grossmann, a similar bijection can be constructed. In this way it can be proved that the

MINLP representation of Kocis and Grossmann represents the same superstructure as represented by the Basic GDP Representation.

4.6. Multiplicity of the MINLP representation

Depending on the actual form of the MINLP problem, multiple local optima, and even multiple global optima may occur. As a consequence of their presence, the optimiser may get into a difficult situation because the objective function does not vary over a domain of non-zero measure. Most of the solution methodologies apply forming continuous (NLP) subproblems combined with either MILP subproblems or tree enumeration. Therefore, making distinction between binary and continuous multiplicities seems useful.

One way of forming NLP subproblems is fixing all the binary values. (For example, this is applied in the outer approximation method.). Binary multiplicity exists if two different NLP subproblems, belonging to differently fixed values of the binary variables, lead to the same optimum. The continuous variables may also take different values.

On the other hand, any NLP subproblem can also have several different optimal solutions with equal objective values. This means that different values of the continuous variables of the NLP, assigned by fixed values of the binary variables, lead to the same optimum. We say that the MINLP problem has continuous multiplicity if one of its NLP problems has this property.

Both the binary and the continuous multiplicity are determined by the form of the MINLP problem. The actual form is somewhat arbitrary; it is in the hand of the engineer or the mathematician. A great extent of multiplicity can be introduced or removed by inapt or efficient formulation. A generally applied way of formulation is assigning binary variables for describing existence or non-existence of units. This conventional methodology is applied in the definition of BMR. If this kind of binary variables is applied and if there is redundancy in the superstructure (irrespectively if it is represented by graph or not) then a great extent of binary multiplicity, originated from structural multiplicity and redundancy, may occur. In some cases this kind of binary multiplicity is evident, in other cases it is not, and in some cases they are even not recognized. This is also manifested in the works of the researchers who tried to eliminate the multiplicity, e.g. Reyes-Labarta and Grossmann (2001), Reneaume et al. (1995), Lelkes et al. (2000). Of course, some sources of binary multiplicity may also be independent of the superstructure.

If a supergraph is structurally redundant (i.e. if there are isomorphic graphs amongst its subgraphs) then BMR has binary multiplicity.

This statement can easily be proved: Let a supergraph R be structurally redundant. Then it has at least two isomorphic subgraphs, R_1 and R_2. Since R_1 and R_2 are isomorphic, they represent the same structure. BMR represents all the subgraphs of R, including R_1 and R_2. R_1 can be assigned by fixing the binary variables to value $\mathbf{z}_1$, and R_2 can be assigned by fixing the binary variables to $\mathbf{z}_2 \neq \mathbf{z}_1$. These different binary values assign different NLP-s, but their optima are equal since R_1 and R_2 represent the same structure. Thus BMR has binary multiplicity.

Conversely, the structural redundancy of the supergraph does not follow from the binary multiplicity of its BMR. It is easy to construct two non-isomorphic graphs and an objective function that leads to the same optimum.

A small, perhaps unrealistic, counter-example is shown in Fig. 4.13. Here the process has a single input and a single output stream, their measure are denoted by the real variables x and y, respectively. The single source unit and the single sink unit are permanent units of the supergraph. The two other units are applied parallel; therefore, either one of them may be omitted, leading to different subgraphs, as is shown in Fig 4.14. Suppose that simultaneous existence of Unit 2 and Unit 3 are prohibited.

The two parallel units are of different type; therefore, these two subgraphs are not isomorphic; they represent different process structures. There is no structural redundancy here. Let the existence or omitting of the parallel units be described by binary variables.

Let the objective function be simply y, and let this objective be determined by the simple formulas shown in Figs. 4.13-4.14, according to which parallel unit is in use. In case of Unit 2: $y=x^2$; in case of Unit 3: $y=2x^2$. In case of $x=0$, y has the same value ($y=0$) in both cases. This is a simple case of binary multiplicity without having structural redundancy.

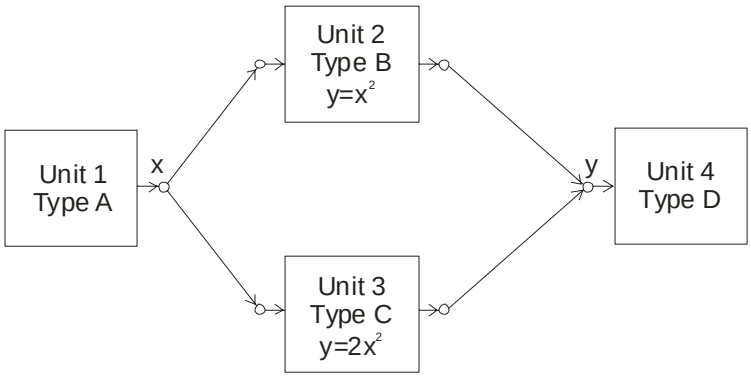


Figure 4.13. Supergraph of the small counterexample

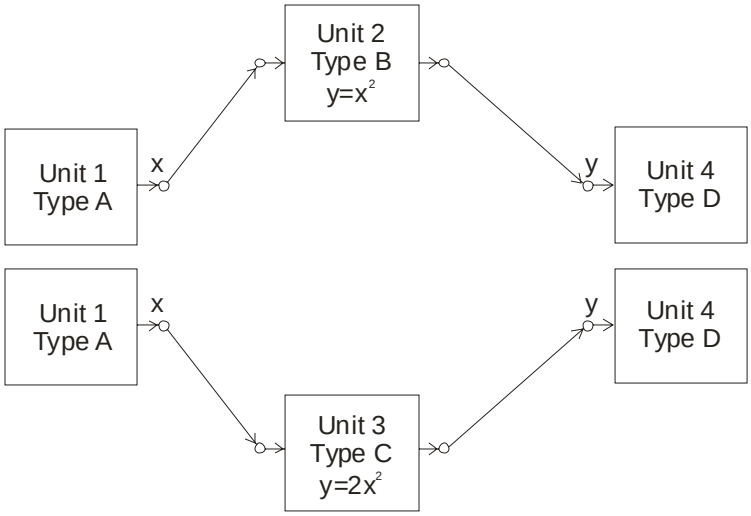


Figure 4.14. Two non-isomorphic feasible subgraphs of the supergraph

As is written by Rév et al. (2005), by-pass multiplicity may occur if not R-graphs are applied. By-pass multiplicity is manifested via continuous multiplicity in the MINLP representation. On the other hand, application of R-graphs does not eliminate all the continuous multiplicity originated from the superstructure. It is an open question if a superstructure eliminating all such multiplicity can always be given.

4.7. Ideality of the MINLP representation

The first task of the engineer, in the process synthesis, is to assign the considered structures. However, a considered structure may be represented by several isomorphic R-graphs, each being a subgraph of the supergraph. Therefore, the engineer has also to assign the set of considered graphs. The set of considered graphs should not include isomorphic pairs. Unfortunately, representation of non-considered graphs by the MINLP representation is not excluded.

The *Ideal MINLP Representation* (IMR) is an MR which represents the considered graphs only, and no other graphs. This is an MR restricted to a surjection ψ from the feasible domain of MR to the feasible domain of BMR. This mapping ψ contains, as a subrelation, $\pi \subseteq \psi$, where π is a bijection between a subset of the feasible domain of MR and the selected set of graphs.

Here a measure of non-ideality is defined as follows: Let the number of considered subgraphs be $n_{cg}>0$; let the number of represented subgraphs be $n_{rg} \geq n_{cg}$, then *the measure of non-*

ideality is the ratio: $NI=(n_{rg} - n_{cg})/n_{cg}$. This measure reaches very high values if the number of isomorphic subgraphs is great.

Although ideal superstructure does not always exist, moreover, in the important cases it usually does not exist, ideal MINLP representation always exists. This existence is shown here by construction. IMR can always be constructed as follows.

First, a BGR and the BMR of the superstructure is constructed. Then, the criteria of considered structures are expressed in logical relationships. Satisfaction of these relationships excludes all the non-considered graphs; thus it enables just the considered graphs to be present in the solution space. These logical relationships include logical relations like 'and', 'or', 'not', 'implication', etc, and logical variables. In most of the cases, the applied logical variables are the logical variables of the BGR, expressing the existence of the units. These logical relationships together form a logical truth function, corresponding to Eq. 4.8. Any truth function can be expressed in conjunctive normal form. How it can be done is described, for example, by Raman and Grossmann (1991). Any conjunctive normal form can be transformed into linear equations using binary variables instead of logical ones. By inserting these linear equations into the BMR, the MR formed this way is an IMR because all the non-considered graphs are excluded by the inserted relations.

4.7.1 Example 4.1 – an ideal MINLP representation

Here the above procedure is demonstrated for constructing IMR of Example 4.1. Earlier, in the problem statement, the criteria of the considered structures have been defined as:

1. Source stream(s) should exist.
2. Product stream(s) should exist.
3. No other outlet streams than the product stream(s) may be present.
4. Unit 2 and Unit 3 should not exist simultaneously.

These criteria can be written via logical variables in the following way:

1. $Z_1 \vee Z_4$ (4.43)
2. Z_6 (4.44)
3. $Z_1 \Leftrightarrow Z_2 \vee Z_3$ (4.45a)

   $Z_2 \vee Z_3 \vee Z_4 \Leftrightarrow Z_5$ (4.45b)

   $Z_5 \Leftrightarrow Z_6$ (4.45c)
4. $\neg\left(Z_2 \wedge Z_3\right)$ (4.46)

Eqs. 4.44 and 4.45c are always true, because Unit 5 and Unit 6 are permanent units. For the same reason Eq. 4.45b can be written in this form:

$$Z_2 \vee Z_3 \vee Z_4 \tag{4.47}$$

The conjunctive normal form of these relations is (see Raman and Grossmann, 1991):

1. $(Z_1 \vee Z_4) \wedge$ (4.48)

3. $(\neg Z_1 \vee Z_2 \vee Z_3) \wedge$ (4.49a)

$(Z_1 \vee \neg Z_2) \wedge$ (4.49b)

$(Z_1 \vee \neg Z_3) \wedge$ (4.49c)

4. $(\neg Z_2 \vee \neg Z_3)$ (4.50)

This conjunctive normal form can be transformed into linear equations using binary variables:

1. $z_1 + z_4 \geq 1$ (4.51)

3. $z_1 - z_2 - z_3 \leq 0$ (4.52a)

$-z_1 + z_2 \leq 0$ (4.52b)

$-z_1 + z_3 \leq 0$ (4.52c)

4. $z_2 + z_3 \leq 1$ (4.53)

An IMR representation is formed simply by appending the BMR with Eqs. 4.51-4.53. On the other hand, R-graphs are used, and it has been shown that the first three criteria are automatically satisfied using R-graph representation. Therefore, Eqs. 4.51-4.52 are automatically satisfied, and only the fourth criterion is to be checked. It follows that only Eq. 4.53 is to be appended to BMR. Then the non-considered graphs are excluded; therefore, the representation becomes ideal.

4.8. Binarily minimal MINLP representation

The problems including integer or binary variables (ILP/MILP/MINLP) are usually more difficult to solve than the problems without them (LP/NLP). The applied form of the equations and the applied methodology of treating the integer variables (type of relaxation) together determine the effectiveness of the solution algorithm and the scale of the solvable problems. Although a great number of integer variables can be successfully dealt with in special cases, they involve serious difficulty in general case. Usually the difficulty (solution time, for example) of solving the problem drastically increases with the number of integer variables. Therefore, decreasing the number of integer variables has a key role in increasing the scale of the solvable problems and decreasing the solution work.

In principle, all the integer variables can be eliminated by an appropriate transformation of the variables and the equations. For example, any binary variable z can be substituted by a continuous variable x with the following extra constraint: $x(x\text{-}1)=0$. In this case the feasible values of x are 0 and 1. On the other hand, such elimination introduces another type of difficulty, i.e. it is usually unsolvable by the commonly applied solvers. Therefore, compromise is to be found between the number of integer variables and the equation forms applied in the problem formulation. It should also be taken into account that the commonly applied solver algorithms usually enable the binary variables appearing in linear members only in the equations.

Here the compromise is suggested that applies the minimum number of binary variables with the constraints that (i) the structural variants are all distinguished by binary variables and (ii) the linearity of the binary members in the equations is maintained.

From here on, the simple case of using the binary variables is considered to distinguish between the structural variants only. With this simplification, the term 'binary variable' will mean 'binary variable applied to make distinction between different structures'.

An MR is called *Binarily Minimal MINLP Representation* (BMMR) if it applies a minimum number of binary variables to make distinction between represented subgraphs.

In mathematical sense it can be formulated as application of the minimum number of binary variables with the condition that an injective (i.e. invertible) mapping can be given from the domain $\mathbb{Z}$ of $\mathbf{z}$ (the binary variables) to the set $\mathbb{R}=\{r_1, r_2, \ldots, r_{n_g}\}$ of the subgraphs. This condition means that any $r_l \in \mathbb{R}$ graph is assigned to only one value of $\mathbf{z}$. If the MR is BMMR, such a mapping with less number of binary variables cannot be given.

It is useful if one knows how many binary variables are to be minimally used. This can be readily given by knowing how many graphs are to be described. An array of n_{bv} binary variables can take $2^{n_{bv}}$ different values. This number is to be at least as great as the number n_g of the R-graphs: $2^{n_{bv}} \geq n_g$. That is, n_{bv} is the smallest whole number that satisfies $n_{bv} \geq \log_2 n_g$. The equality is satisfied only if n_g is exactly a whole power of 2.

In practice, the above set $\mathbb{R}$ of described graphs is to be the set of considered graphs. If n_g is a whole power of 2 then the BMMR is necessarily an IMR because no other graph is described than those in $\mathbb{R}$. However, if n_g is not a whole power of 2 then the number of graphs described by the binary variables can be greater than n_g. For example, let n_g =10; then n_{bv}=4 (because $2^3=8<10<16=2^4$), and $2^{n_{bv}}=2^4=16$. Thus, 1 to 6 additional graphs (numbered as 11, 12, …, 16)

could be described by the binary variables depending on if they are part of the feasible region or not. Should the BMMR not be IMR, it can always be made also an IMR by applying extra constraints that exclude the superfluous graphs from the feasible region.

Binarily minimal MINLP representation can always be constructed. Here we prove this theorem by construction. The method presented here for constructing BMMR serves this purpose only, and should not be considered as the only or the best methodology. Different methodologies can be elaborated, and numerical viewpoints can be taken into account, when the engineer constructs a proper methodology for the particular shape of the problem at hand. The methodology used here as a proof is that follows.

1. The conjunctive normal form (CNF), mentioned in the section above describing how to construct IMR, is to be converted into disjunctive normal form (DNF) where all the logical variables take place in each member of the expression. This can be accomplished in an automatic procedure (see its application for process synthesis, for example, in Raman and Grossmann, 1991). The DNF consists of a series of *clauses* (i.e. brackets) joined with 'or' relation, the brackets containing logical items, with or without the operator 'not', separated with 'and'. The DNF looks like $[(Z_1 \wedge \neg Z_2 \wedge Z_3 \ldots) \vee (\neg Z_1 \wedge Z_2 \wedge Z_3 \ldots) \vee (Z_1 \wedge \neg Z_2 \wedge \neg Z_3 \ldots) \vee \ldots]$ where the Z-s, separated by 'and', are the logical variables of the BGR. Each considered graph r_l is assigned the integer index l ($1 \le l \le n_g$). Each of the *clauses* separated by 'or' corresponds to a considered graph r_l (and, in turn, to a distinct structure, see Brendel et al., 2000). That is, DNF is given in the form:

$$\bigvee_{l=1,2,\ldots,n_g} \Big(\bigwedge_{\forall ip \text{ present}} Z_{ip} \bigwedge_{\forall im \text{ missing}} \neg Z_{im} \Big)_l \tag{4.54}$$

where *ip* are indices of units present in considered graph r_l, whereas *im* are indices of units not present in considered graph r_l.

2. The above l indexes as whole numbers can be expressed in binary form. The minimum number of necessary binary variables is determined by the maximum of these indexes, denoted by n_g. This minimum, as we have seen above, is the smallest whole number n_{bv} that satisfies $n_{bv} \ge \log_2 n_g$. Let the binary digits be described by using the binary variables $\tilde{z}_i$ (i=0, 1, 2,…, n_{bv} -1) so that the index l is expressed as

$$l = 1 + \sum_{i=0}^{n_{bv}-1} \tilde{z}_i \, 2^i \tag{4.55}$$

This expression of a particular number l defines an index set $\mathtt{I1}$ of indices i for which $\tilde{z}_i$ =1, and another index set $\mathtt{IO}$ of indices i for which $\tilde{z}_i$ =0. This definition of index sets depends on the actual value of l. Therefore, these index sets are given as functions of l:

$$\begin{aligned}\mathtt{I1}(l) &= \left\{ i \mid \tilde{z}_i = 1 \text{ in Eq. 4.55} \right\} \\ \mathtt{IO}(l) &= \left\{ i \mid \tilde{z}_i = 0 \text{ in Eq. 4.55} \right\}\end{aligned} \tag{4.56}$$

3. Then a set of integer variables Q_l (l=1, 2, …, n_g) is defined as:

$$Q_l = \sum_{i \in \mathtt{I1}(l)} \left(1 - \tilde{z}_i \right) + \sum_{i \in \mathtt{IO}(l)} \tilde{z}_i \tag{4.57}$$

If the actual value of $\tilde{\mathbf{z}}$ substituted into Eq. 4.55 gives the value l then Q_l=0; otherwise $Q_l \geq 1$.

4. In the next step, the BMR is transformed into BMMR using the above defined $\tilde{\mathbf{z}}$ and Q_l variables instead of **z**. The BMR is formed using Big M technique. The Big M equations of the BMR contains members multiplied by factors of z_i or $(1-z_i)$, see Eqs. 4.13-4.23 in general, and Eqs. 4.29-4.35 of Example 4.1.

First, the n_g equations from Eq. 4.57 (l=1,2,…, n_g) are introduced.

Then for each unit m, it can be easily decided if it is contained by graph l, using the decomposition of the index set of units (or their logical 'presence' variables Z_m) to *ip* and *im*, according to Eq. 4.54.

For each unit m, and for each equation of unit m that *does* contain a factor z_m, this equation is used as many times as many considered graphs contain unit m. In each of these copies the factor z_m is substituted by a unique Q_l, where l is the index of a considered graph that contains unit m. That is, each such l is applied by turn.

For each unit m, and for each equation of unit m that contains a factor $(1-z_m)$, this equation is used as many times as many considered graphs *do not* contain unit m. In each of these copies the factor $(1-z_m)$ is substituted by a unique Q_l, where l is the index of a considered graph that *does not* contain unit m. That is, each such l is applied by turn.

That is, for each unit m, n_g variants of each equation are, in principle, formed. In practice, many of these newly formed equations are redundant, therefore may be omitted.

At this point, i.e. by completing step 4, BMMR is formed. This form of BMMR has the merit that all the equations containing binary variables are linear; i.e. no non-linear equation is added to the original form of BMR.

4.8.1 Example 4.1 – a binarily minimal MINLP representation

Here we demonstrate the above procedure for constructing the BMMR of Example 4.1.

The disjunctive normal form can be accomplished from conjunctive normal form (Eqs. 4.48-4.50) in an automatic procedure. This is Eq. 4.58. Each of the clauses corresponds to a considered graph. For example, the Z_3 logical variable is false in the first clause; all the other logical variables are true. Thus, this clause corresponds to a graph in which all the units exist except Unit 3. This graph is shown in Fig. 4.8a. There are 5 considered graphs, these are presented in Figs. 4.8a-e, and all these graphs are denoted by one clause in DNF. The considered graphs are assigned the integer index j. It is shown in Eq. 4.58.

$$
\begin{array}{lll}
 & \underline{\text{R-graph}} & \underline{l} \\
\left(Z_1 \wedge Z_2 \wedge \neg Z_3 \wedge Z_4 \wedge Z_5 \wedge Z_6\right) \vee & \text{Fig. 4.8a} & 1 \\
\vee \left(Z_1 \wedge Z_2 \wedge \neg Z_3 \wedge \neg Z_4 \wedge Z_5 \wedge Z_6\right) \vee & \text{Fig. 4.8b} & 2 \\
\vee \left(Z_1 \wedge \neg Z_2 \wedge Z_3 \wedge Z_4 \wedge Z_5 \wedge Z_6\right) \vee & \text{Fig. 4.8c} & 3 \\
\vee \left(Z_1 \wedge \neg Z_2 \wedge Z_3 \wedge \neg Z_4 \wedge Z_5 \wedge Z_6\right) \vee & \text{Fig. 4.8d} & 4 \\
\vee \left(\neg Z_1 \wedge \neg Z_2 \wedge \neg Z_3 \wedge Z_4 \wedge Z_5 \wedge Z_6\right) \vee & \text{Fig. 4.8e} & 5
\end{array}
\qquad (4.58)
$$

There are 5 considered graphs, therefore $3 \geq \log_2 5$ binary variables are necessary to generate the binarily minimal MINLP representation. Using the three new binary variables, any integer variable l can be expressed according to Eq. 4.59:

$$l = 1 + \tilde{z}_1 + 2 \cdot \tilde{z}_2 + 4 \cdot \tilde{z}_3 \qquad (4.59)$$

Then the set of integer variables Q_l can be defined as in Eqs. 4.60:

$$
\begin{aligned}
Q_1 &= \tilde{z}_1 + \tilde{z}_2 + \tilde{z}_3 \\
Q_2 &= \left(1 - \tilde{z}_1\right) + \tilde{z}_2 + \tilde{z}_3 \\
Q_3 &= \tilde{z}_1 + \left(1 - \tilde{z}_2\right) + \tilde{z}_3 \\
Q_4 &= \left(1 - \tilde{z}_1\right) + \left(1 - \tilde{z}_2\right) + \tilde{z}_3 \\
Q_5 &= \tilde{z}_1 + \tilde{z}_2 + \left(1 - \tilde{z}_3\right)
\end{aligned}
\qquad (4.60)
$$

For example, the graph shown in Fig. 4.8d is represented by l=4 via Eqs. 4.58-4.60. According to Eq. 4.59, variable l can take the value 4 if $\tilde{\mathbf{z}}$=[1, 1, 0]. It means that Q_4=0 and all the other Q_l–s take positive integer value (Q_1=2; Q_2=1; Q_3=1; Q_5=3).

In the next step, the BMR is transformed into BMMR. Only those equations are changed which contain binary variables (Eqs. 4.26-4.32). The transformed equations of the input streams are given in Eqs. 4.61.

$$0 \le x_2^{in} \le 2.04 Q_l \qquad l = 3,4,5 \tag{4.61a}$$

$$-x_2^{in} \le 2.04 Q_l - \varepsilon \qquad l = 1,2 \tag{4.61b}$$

$$0 \le x_3^{in} \le 1.53 Q_l \qquad l = 1,2,5 \tag{4.61c}$$

$$-x_3^{in} \le 1.53 Q_l - \varepsilon \qquad l = 3,4 \tag{4.61d}$$

x_2^{in} takes positive value if Unit 2 exists. According to Eq. 4.58, Unit 2 exists if l=1 or l=2, and does not exist if l=3, l=4, or l=5. Consider, for example, the case where l=3 and Unit 2 does not exist. Q_3=0 according to Eq. 4.60, and the other Q_l ($l \neq 3$) variables take positive integer value. The three different equations denoted by Eq. 4.61a are detailed in Eqs. 4.62a-c. Variable x_2^{in} is forced in Eq. 4.62a to take the value 0. $Q_4 \ge 1$ and $Q_5 \ge 1$ in the two other Eqs. 4.62b-c; therefore $0 \le x_2^{in} \le M$, where M is a number not smaller than the upper bound (2.04) of x_2^{in}. The two equations of Eq. 4.62b are detailed in Eqs. 4.63a-b: If l=3 then $Q_1 \ge 1$ and $Q_2 \ge 1$; therefore, these equations are satisfied by the assignment x_2^{in}=0.

$$0 \le x_2^{in} \le 2.04 Q_3 \tag{4.62a}$$

$$0 \le x_2^{in} \le 2.04 Q_4 \tag{4.62b}$$

$$0 \le x_2^{in} \le 2.04 Q_5 \tag{4.62c}$$

$$-x_2^{in} \le 2.04 Q_1 - \varepsilon \tag{4.63a}$$

$$-x_2^{in} \le 2.04 Q_2 - \varepsilon \tag{4.63b}$$

Another example is the case where l=1 and Unit 2 exists. Q_1=0 and $Q_{l \neq 1} \ge 1$ in this case. Variable x_2^{in} has to take a positive value not smaller then ε, according to Eq. 4.63a. But none of the other Eqs. 4.62a-c and 4.63b effects the value of x_2^{in}; thus, x_2^{in} can take any value in interval $[\varepsilon, (x_2^{in})^{UP}=2{,}04]$.

Eqs. 4.61c-d work similarly. Variable x_3^{in} is positive if Unit 3 exists and if *l*=3 or 4, according to Eq. 4.58. If *l*=1, 2 or 5 then Unit 3 does not exist, and Eq. 4.61c forces x_3^{in} to take value 0. If *l*=3 or 4 then Unit 3 exists and x_3^{in} can take any value in interval [ε, (x_3^{in})^{UP}=1.53].

Similar equations can be written for the output streams (Eq. 4.64), for the fix and variable costs (Eqs. 4.65-4.66), and for the functions of the output streams and costs (Eqs. 4.67-4.69).

$$\begin{aligned}
&0 \le x_1^{out} \le 3.57 Q_l && l = 5\\
&-x_1^{out} \le 3.57 Q_l - \varepsilon && l = 1,2,3,4\\
&0 \le x_2^{out} \le 1.11 Q_l && l = 3,4,5\\
&-x_2^{out} \le 1.11 Q_l - \varepsilon && l = 1,2\\
&0 \le x_3^{out} \le 1.11 Q_l && l = 1,2,5\\
&-x_3^{out} \le 1.11 Q_l - \varepsilon && l = 3,4\\
&0 \le x_4^{out} \le 1.11 Q_l && l = 2,4\\
&-x_4^{out} \le 1.11 Q_l - \varepsilon && l = 1,3,5
\end{aligned} \tag{4.64}$$

$$\begin{aligned}
&0 \le c_1^{fix} \le 0 Q_l && l = 5\\
&0 \le c_2^{fix} \le 1 Q_l && l = 3,4,5\\
&0 \le c_3^{fix} \le 1.5 Q_l && l = 1,2,5\\
&0 \le c_4^{fix} \le 0 Q_l && l = 2,4
\end{aligned} \tag{4.65}$$

$$\begin{aligned}
&0 Q_l \le c_1^{var} \le 6.42 Q_l && l = 5\\
&0 Q_l \le c_2^{var} \le 5 Q_l && l = 3,4,5\\
&0 Q_l \le c_3^{var} \le 2.31 Q_l && l = 1,2,5\\
&0 Q_l \le c_4^{var} \le 7.78 Q_l && l = 2,4
\end{aligned} \tag{4.66}$$

$$\begin{aligned}
&-1.11 Q_l \le x_2^{out} - \ln\left(x_2^{in} + 1\right) \le 1.11 Q_{l'} && l = 3,4,5;\ l' = 1,2\\
&-1.11 Q_l \le -x_2^{out} + \ln\left(x_2^{in} + 1\right) \le 1.11 Q_{l'} && l = 3,4,5;\ l' = 1,2\\
&-1.11 Q_l \le x_3^{out} - 1.2 \ln\left(x_3^{in} + 1\right) \le 1.11 Q_{l'} && l = 1,2,5;\ l' = 3,4\\
&-1.11 Q_l \le -x_3^{out} + 1.2 \ln\left(x_3^{in} + 1\right) \le 1.11 Q_{l'} && l = 1,2,5;\ l' = 3,4
\end{aligned} \tag{4.67}$$

$$\begin{aligned}
&0 Q_l \le c_1^{fix} - 0 \le 0 Q_l && l = 1,2,3,4\\
&-1 Q_l \le c_2^{fix} - 1 \le 0 Q_l && l = 1,2\\
&-1.5 Q_l \le c_3^{fix} - 1.5 \le 0 Q_l && l = 3,4\\
&0 Q_l \le c_4^{fix} - 0 \le 0 Q_l && l = 1,3,5
\end{aligned} \tag{4.68}$$

$$
\begin{aligned}
&-6.42Q_l \le c_1^{var} - 1.8x_1^{out} \le 6.42Q_l \qquad l = 1,2,3,4\\
&-5Q_l \le c_2^{var} - x_2^{out} \le 5Q_l \qquad l = 1,2\\
&-2.31Q_l \le c_3^{var} - 1.2x_3^{out} \le 2.31Q_l \qquad l = 3,4\\
&-7.78Q_l \le c_4^{var} - 7x_4^{out} \le 7.78Q_l \qquad l = 1,3,5
\end{aligned}
\tag{4.69}
$$

The transformation is the same in case of all the equations. If an equation contains the factor z_m, it is substituted by a set of equations which contains unique Q_l, where l is the index of a considered graph that *does not* contain unit m. Similarly, equations containing the factor $(1\text{-}z_m)$ are substituted by sets of equations, which contain unique Q_l, where l is the index of a considered graph that contains unit m.

As a result of the above transformation, the MINLP representation becomes binarily minimal because it uses the minimal number of binary variables. But it is still not an ideal representation. Combination of three binary variables can take 2^3=8 different values. Thus, variable l can also take 8 different values, according to Eq. 4.59, whereas there are only 5 considered graphs. In the cases of $\tilde{\mathbf{z}}$=[1, 0, 1], [0, 1, 1], and [1, 1, 1], variable l takes value l=6, 7 and 8, respectively. But these values of l do not denote considered graphs, therefore these values have to be excluded from the representation. This can be done by inserting integer cuts in the form of Eqs. 4.70:

$$
\begin{aligned}
&\tilde{z}_1 - \tilde{z}_2 + \tilde{z}_3 \le 1\\
&\tilde{z}_1 + \tilde{z}_2 + \tilde{z}_3 \le 2\\
&-\tilde{z}_1 + \tilde{z}_2 + \tilde{z}_3 \le 1
\end{aligned}
\tag{4.70}
$$

After adding Eqs. 4.70 to the representation, it represents the considered graphs only, thus it is ideal. Since, in the same time, it uses the minimum number of binary variables, this representation is binarily minimal and ideal. Therefore, it is abbreviated as BMIMR.

The above procedure is just one possible method for generating BMIMR; there may be several others. The main advantage of this method is that no more nonlinear equations are added to the representation. On the other hand, relaxation of the Big M equations becomes worse in this way. However, decrease of the total running time is expected because of decreasing the number of the main (outer) iterations.

4.9. Example 4.1 – Computational results

Here the comparison of the different models outlined above according to the computational results with the same example problem hitherto discussed. For this aim, the synthesis problem of Kocis and Grossmann (1987) is studied.

First, the MINLP representation originally given by Kocis and Grossmann ("MR of Kocis") was solved, for comparing the running time. Second, another MINLP representation ("MR") was constructed by first constructing the BMR of the problem and then excluding the redundant and unnecessary equations. Third, the non-considered graphs were excluded from this MR by inserting logical constraints; so that it became ideal ("IMR"). Finally, the binarily minimal and, in the same time, ideal MINLP representation ("BMIMR") was generated via using minimal number of binary variables.

These four representations were solved on a Sun Sparc Station, using GAMS DICOPT++ solver (Brooke et al., 1992; Viswanathan and Grossmann, 1990). All the binary variables were assigned the initial value 0.5 in each case. The stop criterion was set to "STOP 0" because of the presence of non-linear equations. This parameter value results in stopping the iteration only if the MILP subproblem becomes infeasible. The solution times and results are collected in Table 4.2.

Table 4.2. Solution times in Example 4.1

representation	objective value	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)
MR of Kocis	-1.923	8	0.98	0.61	0.37	0.076	0.046
MR	-1.923	16	2.37	1.18	1.19	0.074	0.074
IMR	-1.923	12	1.70	0.86	0.84	0.072	0.070
BMIMR	-1.923	5	0.92	0.44	0.48	0.088	0.096

The second column of Table 4.2 shows the optimal value of the objective function. The number of iterations shows how many main (outer) iterations were done. The solution time is given in CPU sec, and is also broken down to NLP and MILP subproblems. The average NLP and MILP CPU time, per iteration, is given in the last two columns, for comparison.

The same optimum was found in each case. This solution is shown in Fig. 4.15. This graph is represented by l=4 in our BMIMR, as detailed in an earlier section.

16 iterations were needed, using MR, to consider all the feasible MILP subproblems. The number of iterations decreased to 12 as a results of using IMR instead (the non-considered graphs were excluded). The solution time of the NLP and MILP subproblems, and therefore the total solution time as well, decreased by about 28-30 %. Using minimum number of

binary variables (BMIMR) resulted in decreasing the solution time by 46 %. In the same time, a small increase in the solution time per iterations is observed. This effect may be caused by three factors: (1) There are more equations in BMIMR than in IMR; therefore, the NLP-problems became greater. (2) The relaxation is not so good in this case. (3) The number of equations containing binary variables also increased; therefore, the MILP subproblems became more complex. On the other hand, only 6 iterations were necessary in this case to investigate all the feasible MINLP subproblems, thanks to having less number of binary variables. This is, perhaps, why the total solution time decreased.

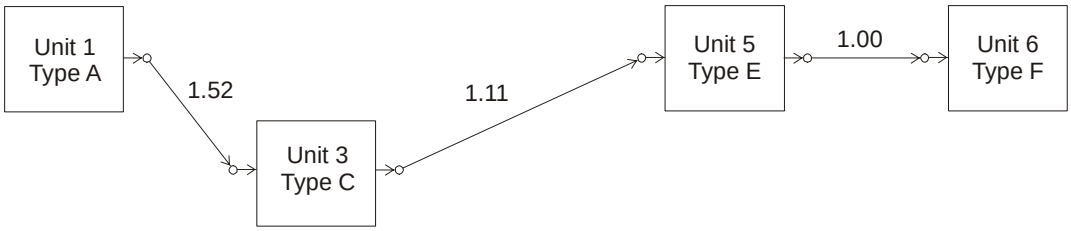


Figure 4.15. The optimal solution of **Example 4.1**

The MINLP representation of Kocis and Grossmann (1987) is binarily minimal because it uses no more than 3 binary variables. We suspect, this is the reason why applying MR of Kocis involved faster solution than MR and IMR did. On the other hand, it is not ideal according to the fourth criterion because it represents graphs including Unit 2 and Unit 3 simultaneously, which are non-considered graphs in our example. The most probable reasons that resulted in 6 % faster solution of our BMIMR than MR of Kocis, even against the somewhat worse relaxation of the MILP subproblems, are the following two properties: (1) BMIMR is, according to its name, ideal representation, i.e. all the non-considered graphs are excluded from the search space. (2) Only some of the variables are given upper bounds in MR of Kocis.

In this example just one method was applied for constructing IMR and BMIMR. The same methodology can be applied to any MINLP problem, but not always will, perhaps, improve the numerical properties of the solution procedure. The number of necessary iterations is decreased to its third in one case, and the solution time is decreased to 39%, even for this rather small example, by idealizing the representation.

4.10. Example 4.2 – Pervaporation system

Here the discussion of the membrane subsystem of an industrial distillation and membrane hybrid system of ethanol dehydration, published by Szitkai et al. (2002), is presented. The aim with this example is to show how idealization of the MINLP representation and reduction of the number of binary variables result in both faster solution and essential increase in the problem scale solvable with the commercial MINLP solvers.

The ethanol dehydration system has two main parts (Fig. 4.16): a distillation column and a pervaporation system. The top product of the column is ethanol-water mixture of near azeotropic composition, and is fed to the first section of a membrane train. The membrane train is a series of several sections each containing a set of parallel connected membrane modules having the same inlet in that section. The retentate streams of the membrane modules are collected; a part is led away as product, whereas the other part is fed to the next section of membranes through a heat exchanger. The permeate streams of the membranes are also collected and mixed to the feed of the distillation column.

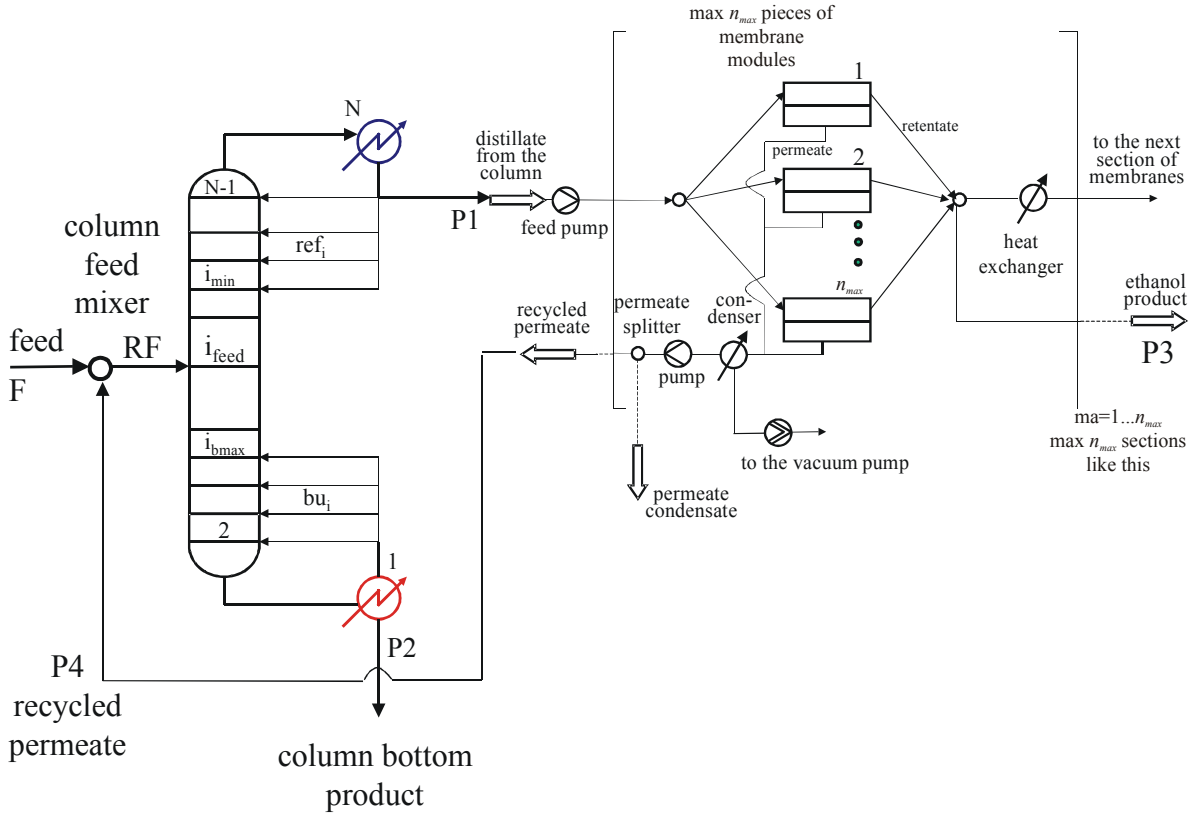


Figure 4.16. Superstructure of the ethanol dehydration system.

In this example the membrane subsystem is considered alone, with given distillate properties. The flowrate of the feed to the membrane subsystem is 50 kg/hr; and it contains 10.05 mol % of water. The target is to reach 4 mol % of water in the final retentate. The problem originally presented by Szitkai et al. (2002) is simplified so that here the flowrates and the concentrations as the only parameters of the flows are considered. The main difference is in the objective function (Eqs. 4.71).

$$c^{var} = \frac{\$775/yr}{3} \cdot \sum_{ma=1}^{n_{max}} UNITS_{ma}$$

$$c^{fix} = \$161.6108/yr \cdot \sum_{ma=1}^{n_{max}} UNITS_{ma} \qquad (4.71)$$

$$C = c^{var} + c^{fix}$$

where $UNITS_{ma}$ is the number of membrane modules in section *ma.*

The first step in solving this problem is defining a superstructure. For this aim, we constraint the membrane system in a "square" shape of scale n_{max}. The system consists of maximum n_{max} number of sections, and in each section maximum the same n_{max} number of membrane modules can be built in. The membrane modules are modelled as if they were all built up from three membrane cells of 1/3 m^3 surface.

The R-graph representation of the superstructure is generated as follows. The supergraph for five membrane sections and five membrane units in each section (a 5 × 5 system) is shown in Fig. 4.17a. There is a source unit 'Feed', and there are two sink units ('Product' and 'Permeate') in the supergraph. Each membrane module is modelled as a unit with one input port and two output ports. Output port 1 is used for releasing the retentate, and output port 2 is for the permeate.

Rév et al. (2005) mentioned that the use of mixer and splitter units is not recommended, because this may lead to redundancy. In order to avoid constructing an unnecessarily complicated supergraph, however, here mixers and splitters were introduced in such a special arrangement that does not introduce by-pass redundancy. This is achieved by leading all the by-pass lines to the final product sink unit 'Product'. After each section there is a mixer/splitter unit, denoted by S, which collects the retentate from the modules in that section, and splits it between the modules of the next section and sink unit 'Product'. Similarly there is a mixer (denoted by M), which collects the permeate of each section, and leads it to the sink unit 'Permeate'. The retentate mixer-splitter unit after the last section is inserted in order to

have the same model for each section. The ratio of one of its output streams to its input stream will be fixed to a constant in each MINLP representation.

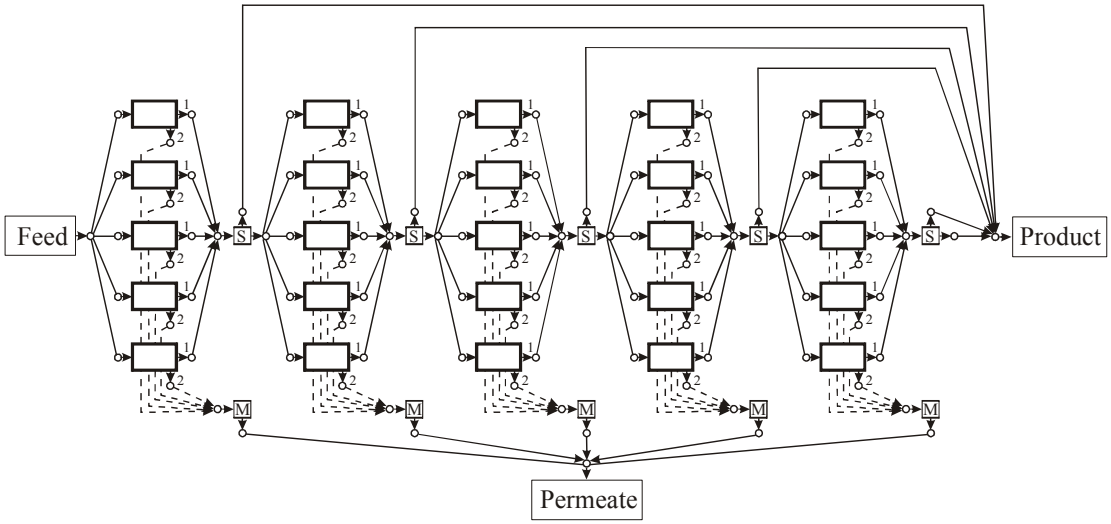


Figure 4.17a. R-graph of the superstructure in Example 4.2

A simplified visualization of the R-graphs is applied from here on. The membrane units are denoted by rectangles, the mixer and mixer-splitter units ('M' and 'S'), as well as the source unit 'Feed' and the sink units 'Permeate' and 'Retentate' are omitted. The corresponding streams are also omitted, and only the main streams are shown. The direction of the streams are also omitted, the edges are always directed from left to right. Shaded rectangles represent built in membrane modules, whereas white rectangles denote empty places of modules. For example, a simplified visualization of the 5×5 supergraph is shown in Fig. 4.17b.
When writing the term 'graph' or 'structure', I always think the full graph version. For example, when calling Fig. 4.17b 'graph' I mean the graph shown in Fig. 4.17a.

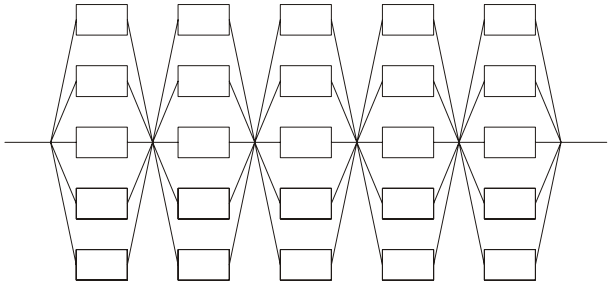


Figure 4.17b. Simplified visualization of R-graph representation

According to engineering experience, the number of parallel modules need not be increased as the material flows downstream. Such a structure would not be optimal. Therefore, only those

structures will be considered that are characterised with non-increasing number of membrane modules in consecutive membrane sections. For example, the graph in Fig. 4.18a represents non-considered structure (Fig. 4.18b) because there are more membrane modules in the second section than in the first section. This structure includes 8 membrane modules. The graph representation of a considered structure (Fig. 4.19b) with the same number of membrane modules (i.e. 8) is shown in Fig. 4.19a.

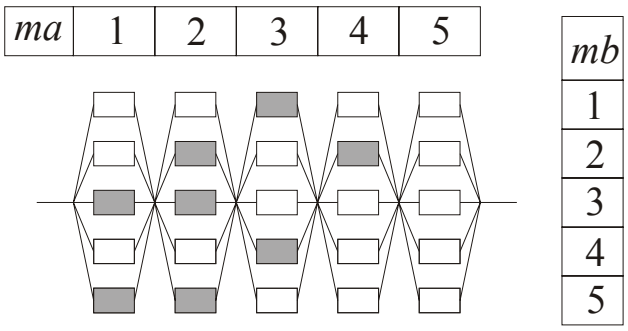


Figure 4.18a Graph of a non-considered structure

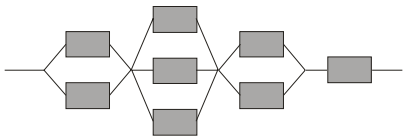


Figure 4.18b A non-considered structure

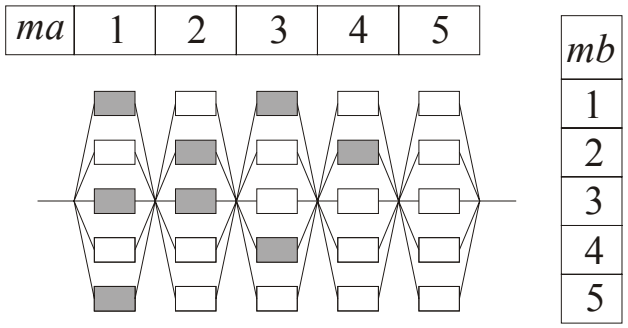


Figure 4.19a Graph of a considered structure

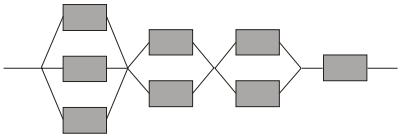


Figure 4.19b A considered structure

4.10.1. Conventional MINLP representation

In this section the construction of an MINLP representation of Example 4.2 according to the conventional approach is presented.

Binary variables are applied to represent the existence of the membrane modules. If $z_{ma,mb}$=1 then place *mb* of section *ma* is occupied by a membrane module, i.e. that module exists, whereas it is not occupied if $z_{ma,mb}$=0. (Here *ma* and *mb* may run from 1 to n_{max}, independently.) The same approach is applied in the BMR representation of R-graphs. However, the MINLP representation used here is not BMR because the redundant and unnecessary equations are excluded. Therefore, this MR is called Conventional MINLP Representation, and denoted by CMR

GAMS DICOPT++ with CONOPT NLP-solver and OSL MINLP-solver on a Sun Sparc Station were used for solving this CMR. The initial value for each binary variable was 0.5. Maximum 50 main iterations, and in all the iterations maximum 50000 iteration steps, were allowed. The CMR has been solved first with n_{max}=5, and then with increased n_{max}. The solution times and results are collected in Table 4.3; the optimal solutions are shown in Figs. 4.20-4.22.

Table 4.3. Results of CMR

n_{max}	cost (USD/yr)	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)	non-ideality
5	4619	7	8.42	0.39	8.03	0.056	1.147	11515
6	4619	3	10.77	0.34	10.43	0.113	3.477	207490
7	4619	3	39.65	0.47	39.18	0.157	13.060	3784425
8	-	-	-	-	-	-	-	69023979

The problem size n_{max}, as an upper limit of both the consecutive membrane sections and of the number of parallel modules in each section, is shown in the first column. The ‘cost’ column shows the total cost found in the optimal solution. The solution time is broken down into NLP and MILP subproblem time; these subtime values are also given as average values related to iterations. The non-ideality of the representation, i.e. how many non-considered graphs are also represented by the CMR, is given in the last column. The solution consists of 11 membrane modules in each case, as is shown in Figs. 4.20-4.22, corresponding to the lines of Table 4.3. For n_{max} =8, and for bigger problems, the solver does not find integer solution within the iteration limit.

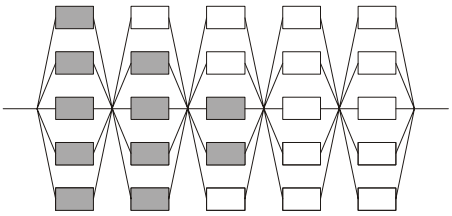


Figure 4.20. Optimal solution with CMR, n_{max} =5

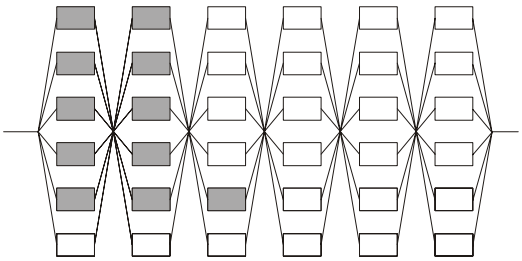


Figure 4.21. Optimal solution with CMR, n_{max} =6

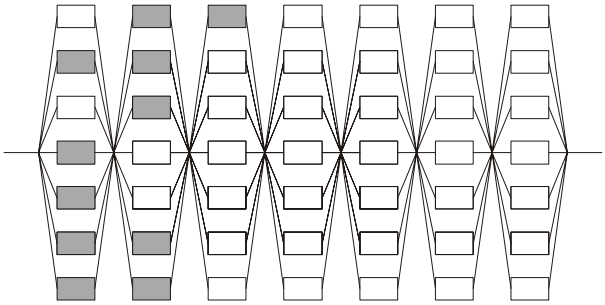


Figure 4.22. Optimal solution with CMR, n_{max} =7

The non-ideality of the representations is incredible high, as is seen in the last column. There are two reasons of this great non-ideality:

(1) There is not any equation in CMR that would force the solution to satisfy the criterion of considered structures, i.e. the non-increasing number of membrane modules in consecutive membrane sections. On the other hand, each solution in Figs. 4.20-4.22 agrees this assumption; therefore, this criterion seems to be right.

(2) CMR represents incredible high number of isomorphic graphs; and most of them are non-considered graphs. Consider, for example, a system with maximum 5 sections, and maximum 5 modules in each section (n_{max} =5). If there is only one membrane module in the actual structure, and if the criterion of non-increasing number of membrane modules in consecutive membrane sections *is satisfied* then the structure can be represented with 5 isomorphic graphs because that single module can be placed in any of the possible places in the first section. This single unit cannot be put in the second or a later section, because in this case there was no

route from the feed to that unit. Two of these 5 graphs are shown in Fig. 4.23. These graphs represent the same structure. It follows that the structural multiplicity of the structure consisting of one membrane is 5 in CMR, in case of n_{max} =5. But only one of these isomorphic graphs is a considered graph, the others are non-considered ones. As the CMR represents all the feasible graphs, it represents the non-considered graphs as well; therefore, CMR is not an ideal representation.

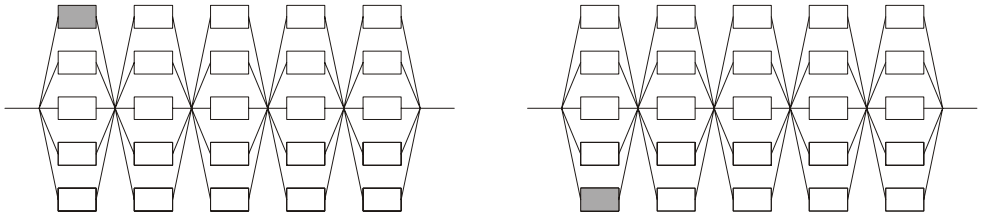


Figure 4.23. Isomorphic graphs

The number of represented graphs (n_{rg}) and that of the considered graphs (n_{cg}) are needed to calculate the non-ideality $NI=(n_{rg}–n_{cg})/n_{cg}$ of a representation. Determining them analytically is a rather complicated task. I am content with approximating non-ideality measures for comparing the problems according to this viewpoint. Therefore, the non-ideality is estimated for n_{max} =2, 3, 4, and a trendline is set in logarithmic plot. The non-ideality of the representation can be estimated for bigger problems, as well, by applying approximating equations of this trendline (Farkas, 2001).

The NLP solution time per iterations increases with the size of the problem (n_{max}) linearly, whereas the MILP solution time per iterations increases exponentially, as is shown in Fig. 4.24.

The linear increase of the NLP solution time per iterations is caused by the increase of the size of the problem and the equation system. The non-ideality (see Table 4.3) and the number of binary variables increase exponentially with the size of the problem; therefore, the search space of the MILP problem also increases exponentially. That is the reason of the exponential increase of the MILP solution time per iterations. The non-ideality increases drastically with increasing problem size, and this is why the problem becomes insolvable in case of 8×8 membrane systems and above.

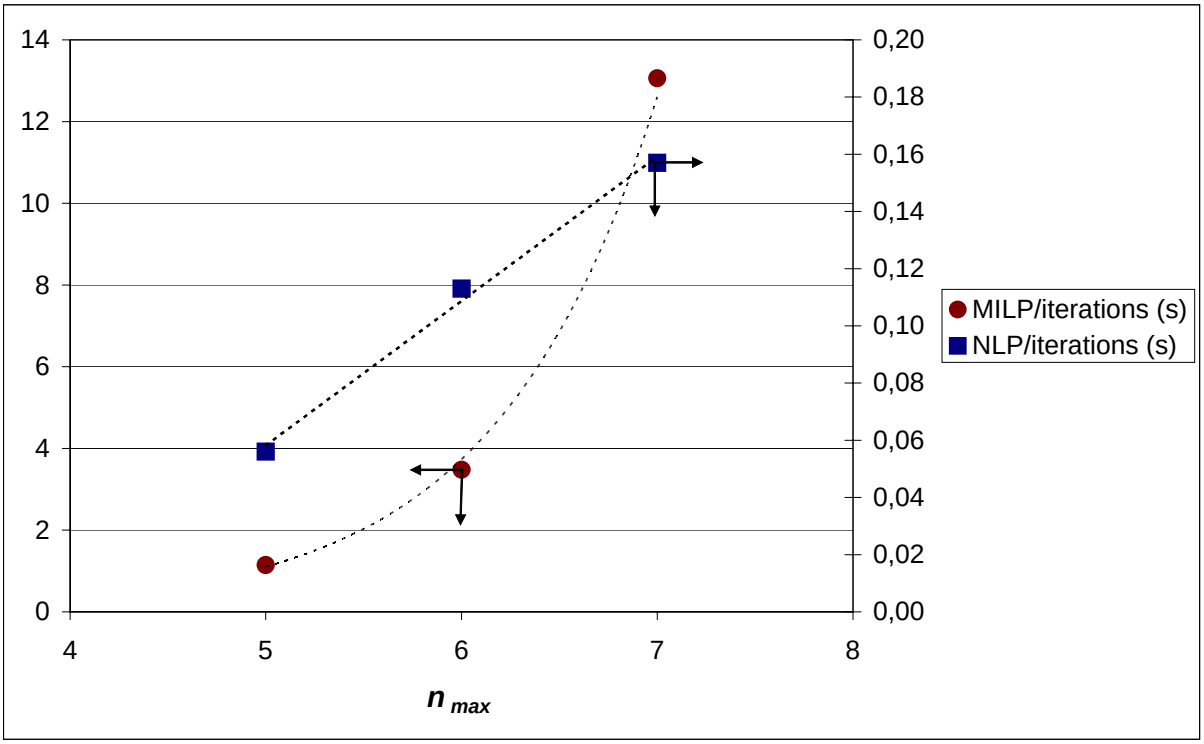


Figure 4.24. Solution times per iterations in CMR

4.10.2. Exclusion of the non-considered structures

Non-ideality can be decreased by excluding the representation of those graphs, from the MINLP representation, that represent non-considered structures. Considered structures are the structures with non-increasing number of membrane modules in consecutive membrane sections. (Refer to Figs. 4.18-4.19, and the corresponding notes in the text.) The search space can be decreased by excluding some of, or all, the, graphs representing non-considered structures

In order to exclude the representation of the graphs of non-considered structures from the representation, Eq. 4.72 is added to CMR. This equation forces the solution to satisfy the criterion of non-increasing number of membrane modules in consecutive membrane sections. This representation is called NSXMR (Non-considered Structures eXcluded MINLP Representation).

$$\sum_{mb=1}^{n_{max}} z_{ma,mb} \geq \sum_{mb=1}^{n_{max}} z_{ma+1,mb} \qquad ma = 1,2,...,n_{max}-1 \tag{4.72}$$

Problems of greater scale are expected to be solvable by applying MR with decreased non-ideality. Therefore, case n_{max}=8, which was infeasible by CMR, was solved using NSXMR.

Solution time values are collected in Table 4.4; the optimal solution is shown in Fig. 4.25. The non-ideality of the representation was estimated in the same way as in the case of CMR.

Table 4.4. Results of NSXMR

n_{max}	cost (USD/yr)	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)	non-ideality
8	4619	3	125.08	0.67	124.41	0.223	41.470	706299
9	4619	-	-	-	-	-	-	6137574

The non-ideality of NSXMR with n_{max} =8 is smaller by two orders of magnitude, comparing to CMR. Even more, it is smaller than the non-ideality of CMR in case of n_{max} =7. That is why the 8×8 problem is solvable using NSXMR.

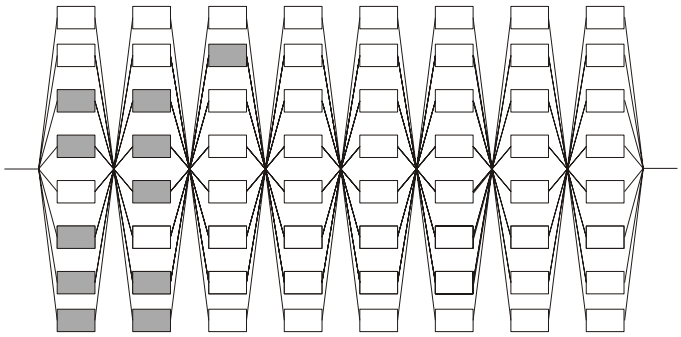


Figure 4.25. Optimal solution with NSXMR, n_{max} =8

Non-ideality of NSXMR with n_{max} =9 is two times greater than that of CMR with n_{max} =7. CMR with n_{max} =7 contains 109 equations and 147 variables, whereas NSXMR with n_{max} =9 consists of 146 equations and 205 variables; i.e., their solutions are approximately of the same difficulty. This is why the 9×9 problem is not solvable using NSXMR.

4.10.3. Ideal MINLP representation

The non-ideality can be further decreased by adding more inequality equations to NSXMR in order to exclude the representation of all the non-considered isomorphic graphs from the representation. Isomorphic graphs represent the same structure; therefore, only one of them is considered, whereas the other graphs are non-considered ones. By excluding these non-considered graphs from the MINLP representation, it becomes ideal.

The representation of non-considered graphs can be excluded from the representation by adding the criterion that the membrane modules should fill in the free places of the actual

section in a queue. In other words: If there are *h* membrane modules in a section, these should be put to the first *h* places. This criterion can be expressed in algebraic form with the help of the binary variables: Each binary variable of a module in a section should not be less than the binary variable of the next module in the same section. This criterion is expressed with Eq. 4.73.

$$z_{ma,mb} \geq z_{ma,mb+1} \quad ma = 1,2,...,n_{max};\ mb = 1,2,...,n_{max} - 1 \tag{4.73}$$

Consider, for illustration, the two isomorphic graphs shown in Fig. 4.28; these graphs represent the same structure. The graph in Fig. 4.28a does not satisfy Eq. 4.73 because, for example, the binary variable of the second module in the first section $z_{1,2}$=0 is smaller than the binary variable of the third module of the same section $z_{1,3}$=1; therefore, this graph is not represented in the Ideal MINLP Representation. The graph in Fig. 4.28b satisfies Eq. 4.73. There is no other graph that would be represented and be isomorphic with this one. This is the only represented graph of all its isomorphic graphs. This applies to all the other represented graphs, as well. The representation of all the isomorphic graphs are excluded from the representation. It follows that the representation becomes ideal (IMR) by inserting Eq. 4.73 to NSXMR.

As the non-ideality of the representation is decreased to 0 (ideal representation), problems of greater scale are expected to be solvable. First the case of n_{max} =9, insolvable by NSXMR, was solved using IMR, and then the size of the problem was increased. The solution times are collected in Table 4.5, and the solutions are shown in Figs. 4.27 and 4.28. The non-idealities are not indicated in Table 4.5 because the representation is ideal, and thus the non-ideality is 0.

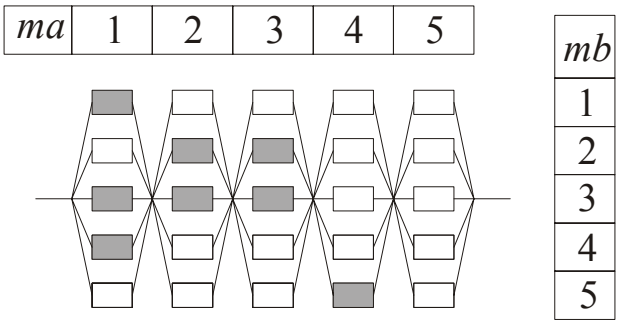


Figure 4.26a. A non-considered graph

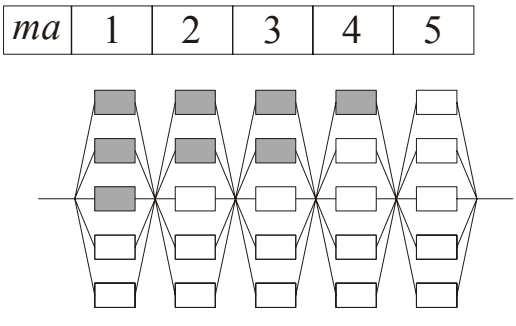


Figure 4.26b. A considered graph

Table 4.5. Results of IMR

n_{max}	cost (USD/yr)	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)
9	4619	4	4.08	0.76	3.32	0.190	0.830
10	4619	3	2.65	0.85	1.80	0.283	0.600
11	-	-	-	-	-	-	-

Solution was found in case of n_{max} =9 and 10; the 11×11 problem was found insolvable with IMR.

The solution time of the NLP subproblems slightly increased because the number of equations and variables were increased with the problem size. On the other hand, the search space and the solution time of the MILP subproblems were decreased drastically, by two orders of magnitude, as a result of excluding the representation of the isomorphic graphs from the representation.

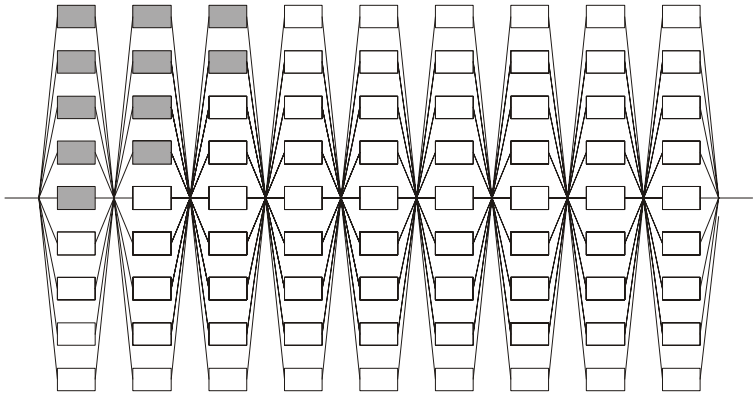


Figure 4.27. Optimal solution with IMR, n_{max} =9

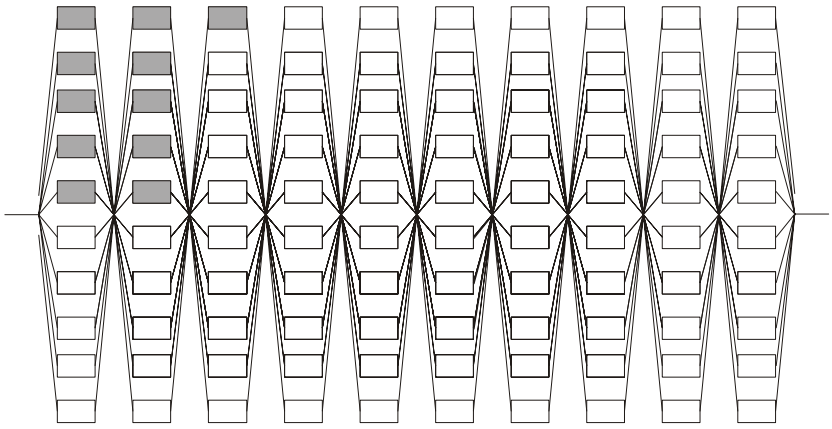


Figure 4.28. Optimal solution with IMR, n_{max} =10

The maximum size of problems solvable with CMR was 7×7=49. The solvable size was increased by one, to 8×8=64, by excluding the representation of the graphs of non-considered structures (using NSXMR). And finally, even the 10×10=100 problem became solvable, and solution time was decreased drastically, as a result of excluding the representation of the isomorphic graphs and using ideal representation (IMR). No further constraints can be added to the representation because the representation is already ideal. Thus, the size of the solvable problems cannot be increased further this way.

4.10.4. Decreased number of binary variables

Further advance is expected by decreasing the number of binary variables. The minimum number of binary variables is suggested to use. However, this minimum number cannot be calculated in this case because the number of represented graphs is not known exactly. We have only an approximation for this minimum. But any kind of decrease in the number of binary variables is expected to enhance the solution.

A possible way to decrease the number of binary variables is using minimum number of binary variables in each section. The number of membrane modules in a section can be expressed as the sum of the binary variables in the section (Eq. 4.74) if either CMR, NSXMR, or IMR is applied:

$$UNITS_{ma} = \sum_{mb=1}^{n_{max}} z_{ma,mb} \qquad ma = 1,2,...,n_{max} \tag{4.74}$$

In order to use minimal number of binary variables in section *ma*, the number of membrane modules is expressed in binary number system, where the binary variable $\tilde{z}_{ma,x}$ denotes the binary digit belonging to 2^{x-1}:

$$UNITS_{ma} = \tilde{z}_{ma,1} + 2 \cdot \tilde{z}_{ma,2} + 4 \cdot \tilde{z}_{ma,3} + 8 \cdot \tilde{z}_{ma,4} \tag{4.75}$$

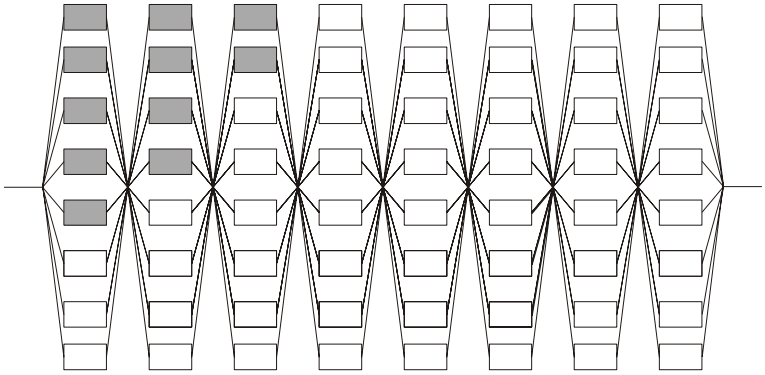
The use of Eq. 4.75 is shown in Fig. 4.29. There are five membrane modules in the first section. This can be expressed with the help of new binary variables using Eq. 4.76. The number of membrane modules in the second and third sections can be calculated similarly.

$$UNITS_1 = \tilde{z}_{1,1} + 2 \cdot \tilde{z}_{1,2} + 4 \cdot \tilde{z}_{1,3} + 8 \cdot \tilde{z}_{1,4} = 1 \cdot 1 + 2 \cdot 0 + 4 \cdot 1 + 8 \cdot 0 = 5 \tag{4.76}$$

Each variable $UNITS_{ma}$ can take integer value from 0 to 15, as a result of Eq. 4.75. But, as maximum n_{max} modules can be built in a section, an upper bound is given to $UNITS_{ma}$:

$$UNITS_{ma} \le n_{max} \tag{4.77}$$

a	1	2	3	4	5	6	7	8



$\tilde{z}_{ma,1}$	1	0	0	0	0	0	0	0
$\tilde{z}_{ma,2}$	0	0	1	0	0	0	0	0
$\tilde{z}_{ma,3}$	1	1	0	0	0	0	0	0
$\tilde{z}_{ma,4}$	0	0	0	0	0	0	0	0
$UNITS_a$	5	4	2	0	0	0	0	0

Figure 4.29. A graph showing the use of **Eq. 4.75**

The graphs of non-considered structures were earlier excluded from the representation by Eq. 4.72. This criterion has to be reformulated because of the new binary variables:

$$UNITS_{ma} \geq UNITS_{ma+1} \qquad ma = 1,2,...,n_{max} - 1 \tag{4.78}$$

Eq. 4.73 applied to exclude the isomorphic graphs from representation when using IMR, is not necessary when the new binary variables are used because only the number of membrane modules in a section is determined by these variables, and not the actual placement of the modules themselves. Therefore, isomorphic graphs cannot be represented, and Eq. 4.73 is omitted.

By using Eqs. 4.75, 4.77, and 4.78, the representation becomes ideal because only considered graphs are represented. This representation is thus ideal and is characterised with a decreased number of binary variables. Therefore this is referred to as 'Binarily Decreased Ideal MR' (BDIMR).

In case of n_{max} =11, 121 binary variables were used in IMR, but the optimum could not be found. Using BDIMR, new binary variables are assigned to all sections; therefore, 4×11=44 binary variables are used. Thus, the number of binary variables is decreased almost to its fourth.

Approximately 3.99E+8 considered graphs are potentially present in a 16×16 membrane system. For constructing binarily minimal representation, $\log_2(3.99E+8) = 28.58 \leq 29$ binary

variables ought to be used. Instead, n_{max}×5=90 binary variables are used in BDIMR. The minimal number of binary variables is computed using an estimate from below; therefore, it is assumed that our BDIMR is not a binarily minimal representation.

BDIMR was first applied to a n_{max} =11 membrane system. The size of the problem was then increased. In case of n_{max} =16, Eq. 4.75 is extended so that $UNITS_{ma}$ can take the value above 15:

$$UNITS_{ma} = \tilde{z}_{ma,1} + 2 \cdot \tilde{z}_{ma,2} + 4 \cdot \tilde{z}_{ma,3} + 8 \cdot \tilde{z}_{ma,4} + 16 \cdot \tilde{z}_{ma,5} \tag{4.79}$$

The results are shown in Figs. 4.30 – 4.31, the solution times are collected in Table 4.8. From n_{max} =17 an on, the problem is insolvable.

Table 4.8. Results of BDIMR

n_{max}	cost (USD/yr)	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)
11	4619	4	6.60	2.42	4.18	0.605	1.045
12	4619	4	7.84	2.69	5.15	0.673	1.288
13	4619	4	8.71	3.41	5.30	0.853	1.325
14	4619	3	6.88	4.37	2.51	1.457	0.837
15	4619	3	8.15	5.83	2.32	1.943	0.773
16	4619	4	13.28	6.63	6.65	1.658	1.663
17	-	-	-	-	-	-	-

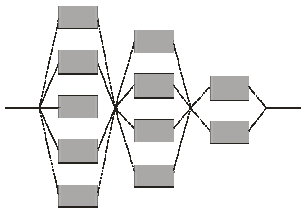


Figure 4.30. Optimal solution with BDIMR, n_{max} =11, 12, 13, 16

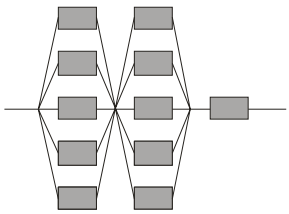


Figure 4.31. Optimal solution with BDIMR, n_{max} =14, 15

The NLP solution time increases with the problem size. In case of n_{max} =14 and 15, the NLP solution time per iteration is a bit greater than expected because of the smaller number of iterations. The MILP solution time increases almost linearly with a very small slope, except in the cases n_{max} =14 and 15, where MILP time decreased almost to its third.

4.10.5. Comparison of the representations

Four MINLP representations were applied to the same pervaporation system: a conventional representation (CMR) that is similar to the basic representation BMR, but the redundant and unnecessary equations were already excluded, a representation excluding the representation of the non-considered structures (NSXMR), an ideal one (IMR), and an ideal and binarily decreased representation (BDIMR). Each representation is an improvement of the preceding one.

Table 4.9. Summary of the computation results for Example 4.2

MINLP repr.	n_{max}	cost (USD/yr)	number of iterations	solution time (s)	NLP (s)	MILP (s)	NLP/it. (s)	MILP/it. (s)	non-ideality
CMR	5	4619	7	8.42	0.39	8.03	0.056	1.147	11515
	6	4619	3	10.77	0.34	10.43	0.113	3.477	207490
	7	4619	3	39.65	0.47	39.18	0.157	13.060	3784425
	8	-	-	-	-	-	-	-	69023979
NSXMR	8	4619	3	125.08	0.67	124.41	0.223	41.470	706299
	9	4619	-	-	-	-	-	-	6137574
IMR	9	4619	4	4.08	0.76	3.32	0.190	0.830	0
	10	4619	3	2.65	0.85	1.80	0.283	0.600	0
	11	-	-	-	-	-	-	-	0
BDIMR	11	4619	4	6.60	2.42	4.18	0.605	1.045	0
	12	4619	4	7.84	2.69	5.15	0.673	1.288	0
	13	4619	4	8.71	3.41	5.30	0.853	1.325	0
	14	4619	3	6.88	4.37	2.51	1.457	0.837	0
	15	4619	3	8.15	5.83	2.32	1.943	0.773	0
	16	4619	4	13.28	6.63	6.65	1.658	1.663	0
	17	-	-	-	-	-	-	-	0

The solution time data and results of all the MINLP representations are summarized in Table 4.9. With the exception of case n_{max} =5, the number of iterations is either 3 or 4 in all the cases.

The NLP solution time per iterations (Fig. 4.32) increases exponentially with the problem size. This is caused most probably by the increasing number of equations. The number of binary variables does not have any effect on the NLP solution time because the binary values are fixed in the NLP subproblems.

The MILP solution time per iterations increases exponentially with the problem size in case of CMR and NSXMR. Data for CMR are shown, but the point of NSXMR is not represented in Fig. 4.33 because of its high value. The reason of this exponential increase lies in the exponential increase in the number of non-considered but represented isomorphic graphs. The representation of the isomorphic graphs are excluded from the representation using IMR and

BDIMR, and the increase of MILP solution time became linear. The solution time with BDIMR at n_{max} =16 (including maximum 256 membrane modules) is comparable with the solution time with CMR at n_{max} =5 (including maximum 25 membrane modules).

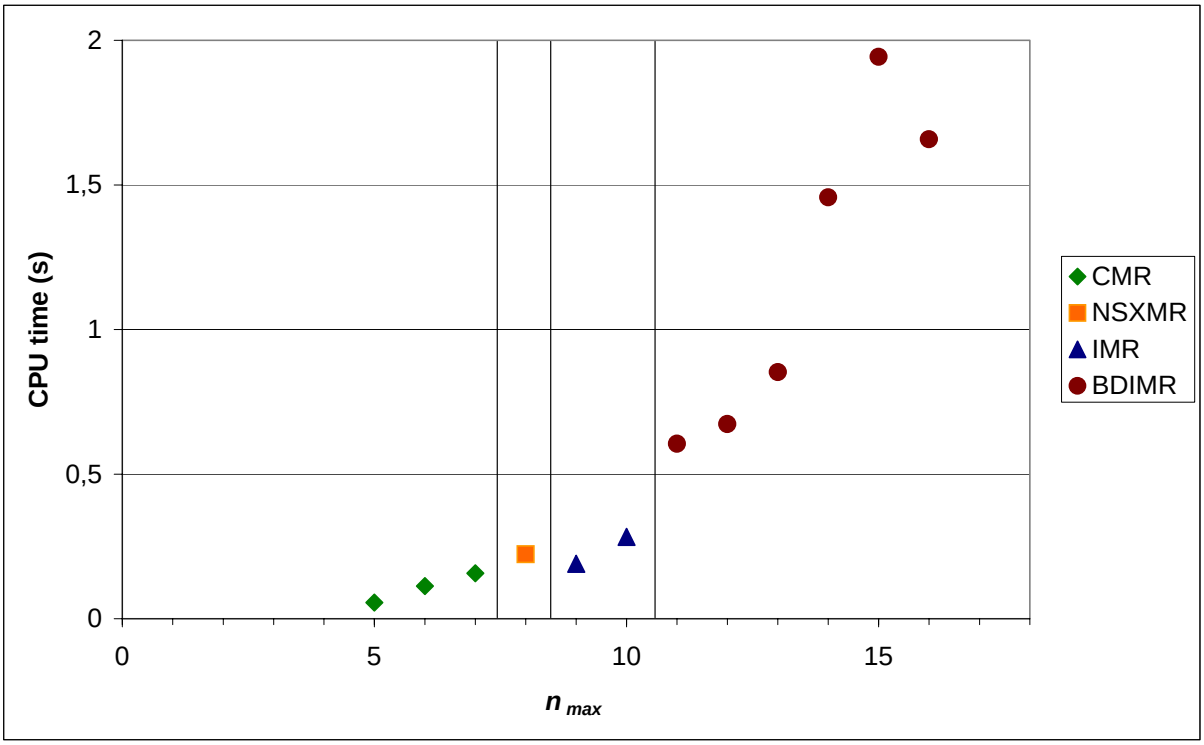


Figure 4.32. NLP solution times per iterations

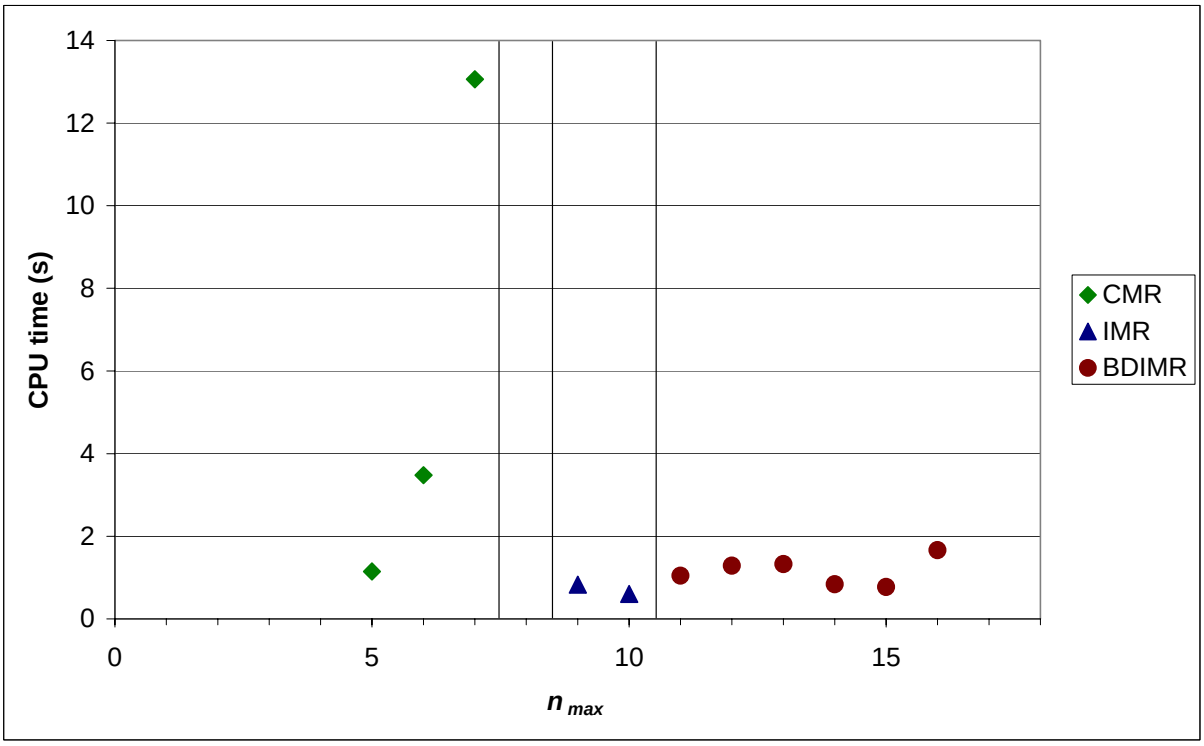


Figure 4.33. MILP solution times per iterations

4.11. Summary

An automated procedure for generating the Basic GDP Representation and the Basic MINLP Representation has been presented.

First the Basic GDP Representation is formulated, based on the R-graph representation of the superstructure. BGR consists of the constraints of the units as logical expressions, the balances of the input and output ports, and the objective function.

The Basic MINLP Representation is formed automatically from BGR by using binary variables instead of the logical ones. BMR can be a reference representation of different MR-s; it is generated automatically from the supergraph, and it represents all the subgraphs of the supergraphs. An MR represents the supergraph if a bijective mapping can be given between a subset of the feasible region of MR and the feasible region of BMR.

This procedure is presented through a small synthesis example.

The chance of finding the global optimum can be increased by excluding the non-considered graphs from the representation, and by decreasing the number of binary variables.

An MINLP Representation is ideal if it represents the considered graphs only. It can be generated by excluding the representation of the non-considered graphs, by applying constraints including logical or binary variables only.

The number of binary variables can usually be decreased by reformulating the MINLP representation. The MR is binarily minimal if it uses the minimum number of binary variables to make distinction between different structures. It means the use of n_{bv} number of binary variables in case of n_g number of represented subgraphs, where n_{bv} is the lowest integer number, which satisfies $2^{n_{bv}} \geq n_g$.

The methods and definitions are demonstrated on a small synthesis problem, and on an industrial example. It is shown that idealization of MR and the decrease of the number of binary variables result in lower computational time, and greater maximum solvable problem size.

5. R-graph-based superstructure and MINLP model for distillation column synthesis

In the previous chapter, it was shown that an MINLP model can be improved by idealization, and by decreasing the number of binary variables. It was performed by adding new constraints containing only new binary variables, and by transforming the old constraints and old binary variables. The results demonstrated that, using these modifications, the solution time can be decreased, and greater problems become solvable.

However, not only in the step of formulating the MINLP representation can this improvement be performed. Sometimes even the superstructure can be generated in a way that the formulation of ideal and/or binarily minimal MINLP model does not need extra transformation.

In this chapter, an R-graph based new superstructure, and MINLP representation, are presented for distillation column synthesis. During the generation of the superstructure and the MINLP, the results of the previous chapter are used.

5.1. R-graph representation of the superstructure

The R-graph superstructure of a conventional distillation column containing maximum 15 equilibrium trays is shown in Fig. 5.1. The equilibrium stages are represented by shaded ovals, and the units by rectangles. The small circles represent input or output ports of the units. The edges, start from output ports and end in input ports represent streams. The edges pointing to top denote the heading direction of the vapor flows, the edges directed down represent the liquid streams. The hatched squares beside the units containing equilibrium stages represent the so-called vapor and liquid transport units; they have transmission roles only. There are one source unit (Feed) and two sink units (Dist and Bot). These sink units represent the products. The reboiler (Reb) and the condenser (Cond) units have two output ports each; by this way, we are able to calculate the reflux and reboil ratio with equations belonging only to the particular units.

In R-graph representation a unit is permanent, by definition, if all the sub-R-graphs contain it. Feed, Dist, Bot, Cond, Reb, and the feed stage, are permanent units according to this definition; all the others are conditional.

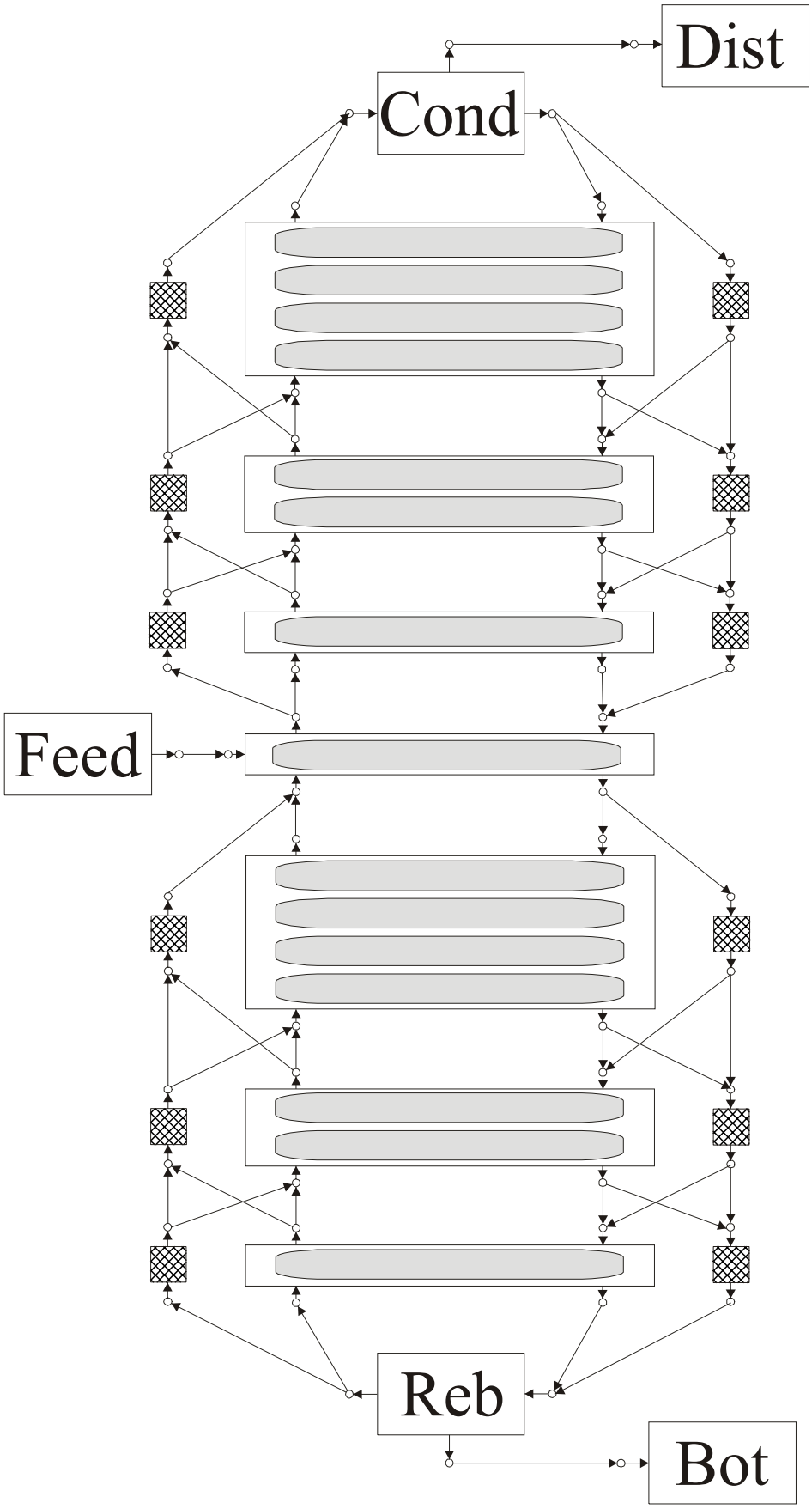


Figure 5.1. R-graph representation of the superstructure

The number of the equilibrium stages in the k^{th} unit of a column section (calculated bottom up) is 2^{k-1}. This arrangement is applied in order to facilitate forming binarily minimal MINLP representation, as explained in section 5.6.

The vapor stream starting e.g. from the vapor output port of the first unit (containing one equilibrium stage) above the reboiler can go to the vapor input port of the second unit, and/or to the input port of the vapor transfer unit shown left to the second unit above the reboiler (this is the 'second vapor transfer unit'). The vapor input port of the third unit can receive vapor stream from the vapor output port of the second unit and/or from the output port of the second vapor transfer unit. My intention with this arrangement is to provide a tool for by-passing the second unit, and to by-passing any conditional unit in the column in this way. The final target is restricting the set of structures to enable only either the conditional unit in the column without the corresponding vapor and liquid transfer units, or to both the corresponding transfer units without the conditional unit in the column. This restriction will be included in the MINLP model, but is not provided in the supergraph representation. Neither is this restriction included in the basic GDP representation, as is discussed in the next section.

5.2. Basic GDP Representation

The Basic GDP Representation is generated according to the R-supergraph (R-graph of the superstructure). It contains the unit relations, the balances of the ports, and the objective function. No additional constraints are added to the GDP model, therefore it represents all the feasible sub-R-graphs of the R-supergraph, and it uses distinct logical variables to each conditional units.

The unit relations of the source unit (Feed; Eq. 5.1) contain the calculation of the molar feed flows from the total feed flow and the feed compositions, and the calculation of the molar enthalpies of the feed from the temperature of the feed. The unit relations of the sink units (Bot, Dist; Eqs. 5.2-5.3) contain only the trivial positiveness constraints. The unit relations of the reboiler (Eq. 5.4) and the condenser (Eq. 5.5) include the heat and component balances. The purity and recovery constraints are specified here, and the variable cost of the column are also calculated here. The unit relations of the fixed feed stage unit (Eq. 5.6) contain the MESH equations, and the fix cost of this stage. The column diameter is calculated in the reboiler, in the condenser and in the feed stage. The greatest among these diameters is the real column diameter.

$$\begin{bmatrix} F_c = Feed \cdot x_c^{Feed} & c \in \mathbf{C} \\ hF_c = \Phi_c^h\left(T^{Feed}\right) & c \in \mathbf{C} \end{bmatrix} \tag{5.1}$$

$$\left[Dis_c > 0 \qquad c \in \mathbf{C} \right] \tag{5.2}$$

$$\left[Bot_c > 0 \qquad c \in \mathbf{C} \right] \tag{5.3}$$

$$\begin{bmatrix} V_c^{in,C} = L_c^{out,C,L} + L_c^{out,C,D} & c \in \mathbf{C} \\ QC = \sum_{c \in \mathbf{C}} \left(V_c^{in,C} \cdot hV_c^{in,C} - \left(L_c^{out,C,L} + L_c^{out,C,D} \right) \cdot hL_c^{out,C} \right) & \\ hL_c^{out,C} = \Phi_c^h\left(T^C\right) & c \in \mathbf{C} \\ x_c^C = \dfrac{V_c^{in,C}}{\sum_{c \in \mathbf{C}} V_c^{in,C}} & c \in \mathbf{C} \\ x_c^C = \dfrac{L_c^{out,C,L}}{\sum_{c \in \mathbf{C}} L_c^{out,C,L}} & c \in \mathbf{C} \\ x_c^C = \dfrac{L_c^{out,C,D}}{\sum_{c \in \mathbf{C}} L_c^{out,C,D}} & c \in \mathbf{C} \\ Ref = \dfrac{\sum_{c \in \mathbf{C}} L_c^{out,C,L}}{\sum_{c \in \mathbf{C}} L_c^{out,C,D}} & \\ x_c^C \geq \tau_c^D & c \in \mathbf{C} \\ x_c^C \leq \lambda_c^D & c \in \mathbf{C} \\ L_c^{out,C,D} \geq \xi_c^D \cdot F_c & c \in \mathbf{C} \\ DC \geq \Phi^D\left(T^C, P^C, \mathbf{V}^{in,C}\right) & \\ c^C = \Phi^{cC}\left(QC\right) & \end{bmatrix} \tag{5.4}$$

$$
\begin{bmatrix}
L_c^{in,R} = L_c^{out,R} + V_c^{out,R} & c \in C \\
QR = \sum_{c \in C} \left(L_c^{in,R} \cdot hL_c^{in,R} - L_c^{out,R} \cdot hL_c^{out,R} - V_c^{out,R} \cdot hV_c^{out,R} \right) & \\
hL_c^{out,R} = \Phi_c^h \left(T^R \right) & c \in C \\
hV_c^{out,R} = \Phi_c^H \left(T^R \right) & c \in C \\
x_c^R = \dfrac{L_c^{in,R}}{\sum_{c \in C} L_c^{in,R}} & c \in C \\
x_c^R = \dfrac{L_c^{out,R}}{\sum_{c \in C} L_c^{out,R}} & c \in C \\
x_c^R = \dfrac{V_c^{out,R}}{\sum_{c \in C} V_c^{out,R}} & c \in C \\
x_c^R \geq \tau_c^R & c \in C \\
x_c^R \leq \lambda_c^R & c \in C \\
L_c^{out,R} \geq \xi_c^R \cdot F_c & c \in C \\
DC \geq \Phi^D \left(T^R, P^R, \mathbf{V}^{out,R} \right) & \\
c^R = \Phi^{cR} \left(QR \right) &
\end{bmatrix}
\tag{5.5}
$$

$$
\begin{bmatrix}
V_c^{in,F} + L_c^{in,F} + F_c^{in,F} = V_c^F + L_c^F & c \in C \\
L_c^{out,F} = L_c^F & c \in C \\
V_c^{out,F} = V_c^F & c \in C \\
L_c^F = LIQ^F \cdot x_c^F & c \in C \\
V_c^F = VAP^F \cdot y_c^F & c \in C \\
\sum_{c \in C} \left(V_c^{in,F} \cdot hV_c^{in,F} + L_c^{in,F} \cdot hL_c^{in,F} + F_c^{in,F} \cdot hF_c^{in,F} \right) = \sum_{c \in C} \left(V_c^F \cdot hV_c^F + L_c^F \cdot hL_c^F \right) & \\
hV_c^F = \Phi_c^H \left(T^F \right) & c \in C \\
hL_c^F = \Phi_c^h \left(T^F \right) & c \in C \\
\sum_{c \in C} x_c^F = 1 & \\
\sum_{c \in C} y_c^F = 1 & \\
f_c^{L,F} = f_c^{V,F} & c \in C \\
f_c^{L,F} = \Phi_c^{fL} \left(\mathbf{x}^F, T^F, P^F \right) & c \in C \\
f_c^{V,F} = \Phi_c^{fV} \left(\mathbf{y}^F, T^F, P^F \right) & c \in C \\
DC \geq \Phi^D \left(T^F, P^F, \mathbf{V}^{in,F} \right) & \\
c^F = \Phi^{cF} \left(DC \right) &
\end{bmatrix}
\tag{5.6}
$$

The unit relations of the conditional units contain two sets of equations with logical 'or' ($\vee$) relation between them. The first set of equations are taken into consideration when the unit exists in a particular structure, in this case the logical variable of the unit is true (Z). If the unit does not exist in a structure, then the other set of equations has to be satisfied; in this case the logical variable of the unit is false ($\neg Z$).

The unit relations of the third unit in a column section (containing four equilibrium stages) are shown in Eq. 5.7. When the unit exists, then the MESH equations of all the included equilibrium stages and the calculation of the fix cost related to this unit are taken into consideration. The component material and enthalpy balances are different for the first (lowest), the last (uppest) and the inner equilibrium stages, because the inlet vapor/liquid stream of the unit ($V_c^{in,s,k}$ / $L_c^{in,s,k}$) goes directly to the first/last equilibrium stage of the unit, and the outlet liquid/vapor stream ($V_c^{out,s,k}$ / $L_c^{out,s,k}$) leaves from the first/last equilibrium stage of the unit.

If the unit does not exist in a structure, then all the component flowrate, enthalpy and cost variables related to the unit take zero value. It means, that there are no inlet and outlet. The inner variables of the unit (e.g. mole fractions and fugacities) can take any value, they do not have any effect to the solution, because all the inlet and outlet variables take zero value.

The unit relations of units containing more than four equilibrium stages are similar to Eq. 5.7. The only difference is, that the set of equilibrium stages is not `J4` but `Jn`, where `n` is the number of stages in the unit.

The unit relations of the second unit in a column section containing two equilibrium stages have two differences from Eq. 5.7. The first is, that the set of equilibrium stages is `J2` instead of `J4`. The second difference is, that in that unit there are only two stages (a first and a last); therefore, the component material balance and the enthalpy balance of the inner stages are missing.

The unit relations of the first unit in a column section containing only one equilibrium stages can be seen in Eq. 5.8. As there is only one stage, there is only one component material balance and one enthalpy balance. The set of equilibrium stages in the unit, `J1` contains only one element, but the set is used because of the analogy with other units containing more equilibrium stages.

The unit relations of the liquid (Eq. 5.9) and vapor (Eq. 5.10) transport units also contain two sets of equations with a logical 'or' between them. When a transport unit exists ($Z^V_{s,k}$ or $Z^L_{s,k}$ is true) the inlet stream goes through the unit without any change in concentration or enthalpy. When a transport unit does not exist ($\neg Z^V_{s,k}$ or $\neg Z^L_{s,k}$) all the inlet and outlet variables take zero value; i.e., there are no inlet or outlet streams at all in that case.

$$
\begin{bmatrix}
Z_{s,k} & \\
\wedge & \\
V^{in}_{c,s,k} + L_{c,j-1,s,k} = V_{c,j,s,k} + L_{c,j,s,k} & j \in \mathsf{J4}^{first} \\
V_{c,j-1,s,k} + L_{c,j+1,s,k} = V_{c,j,s,k} + L_{c,j,s,k} & j \in \mathsf{J4}^{inner} \\
V_{c,j-1,s,k} + L^{in}_{c,s,k} = V_{c,j,s,k} + L_{c,j,s,k} & j \in \mathsf{J4}^{last} \\
L^{out}_{c,s,k} = L_{c,j,s,k} & j \in \mathsf{J4}^{first} \\
V^{out}_{c,s,k} = V_{c,j,s,k} & j \in \mathsf{J4}^{last} \\
L_{c,j,s,k} = LIQ_{j,s,k} \cdot x_{c,j,s,k} & j \in \mathsf{J4} \\
V_{c,j,s,k} = VAP_{j,s,k} \cdot y_{c,j,s,k} & j \in \mathsf{J4} \\
\sum_{c\in\mathsf{C}} \left(V^{in}_{c,s,k} \cdot hV^{in}_{c,s,k} + L_{c,j+1,s,k} \cdot hL_{c,j+1,s,k} \right) = \sum_{c\in\mathsf{C}} \left(V_{c,j,s,k} \cdot hV_{c,j,s,k} + L_{c,j,s,k} \cdot hL_{c,j,s,k} \right) & \\
& j \in \mathsf{J4}^{first} \\
\sum_{c\in\mathsf{C}} \left(V_{c,j-1,s,k} \cdot hV_{c,j-1,s,k} + L_{c,j+1,s,k} \cdot hL_{c,j+1,s,k} \right) = \sum_{c\in\mathsf{C}} \left(V_{c,j,s,k} \cdot hV_{c,j,s,k} + L_{c,j,s,k} \cdot hL_{c,j,s,k} \right) & \\
& j \in \mathsf{J4}^{inner} \\
\sum_{c\in\mathsf{C}} \left(V_{c,j-1,s,k} \cdot hV_{c,j-1,s,k} + L^{in}_{c,s,k} \cdot hL^{in}_{c,s,k} \right) = \sum_{c\in\mathsf{C}} \left(V_{c,j,s,k} \cdot hV_{c,j,s,k} + L_{c,j,s,k} \cdot hL_{c,j,s,k} \right) & \\
& j \in \mathsf{J4}^{last} \\
hL_{c,j,s,k} = \Phi^h_c \left(T_{j,s,k} \right) & j \in \mathsf{J4} \\
hV_{c,j,s,k} = \Phi^H_c \left(T_{j,s,k} \right) & j \in \mathsf{J4} \\
hL^{out}_{c,s,k} = hL_{c,j,s,k} & j \in \mathsf{J4}^{first} \\
hV^{out}_{c,s,k} = hV_{c,j,s,k} & j \in \mathsf{J4}^{last} \\
\sum_{c\in\mathsf{C}} x_{c,j,s,k} = 1 & j \in \mathsf{J4} \\
\sum_{c\in\mathsf{C}} y_{c,j,s,k} = 1 & j \in \mathsf{J4} \\
f^L_{c,j,s,k} = f^V_{c,j,s,k} & j \in \mathsf{J4} \\
f^L_{c,j,s,k} = \Phi^{fL}_c \left(\mathbf{x}_{j,s,k}, T_{j,s,k}, P_{j,s,k} \right) & j \in \mathsf{J4} \\
f^V_{c,j,s,k} = \Phi^{fV}_c \left(\mathbf{y}_{j,s,k}, T_{j,s,k}, P_{j,s,k} \right) & j \in \mathsf{J4} \\
c_{s,k} = \Phi^{c3} \left(DC \right) &
\end{bmatrix}
\vee
\begin{bmatrix}
\neg Z_{s,k} & \\
\wedge & \\
L^{in}_{c,s,k} = 0 & \\
V^{in}_{c,s,k} = 0 & \\
L^{out}_{c,s,k} = 0 & \\
V^{out}_{c,s,k} = 0 & \\
L_{c,j,s,k} = 0 & j \in \mathsf{J4} \\
V_{c,j,s,k} = 0 & j \in \mathsf{J4} \\
hL_{c,j,s,k} = 0 & j \in \mathsf{J4} \\
hV_{c,j,s,k} = 0 & j \in \mathsf{J4} \\
hL^{in}_{c,s,k} = 0 & \\
hV^{in}_{c,s,k} = 0 & \\
hL^{out}_{c,s,k} = 0 & \\
hV^{out}_{c,s,k} = 0 & \\
c_{s,k} = 0 &
\end{bmatrix}
$$

$$c \in \mathsf{C}, s \in \mathsf{S}, k \in \{k \in \mathsf{K} \mid k = 3\} \qquad (5.7)$$

$$\begin{bmatrix} Z_{s,k} \\ \wedge \\ V^{in}_{c,s,k} + L^{in}_{c,s,k} = V_{c,j,s,k} + L_{c,j,s,k} \\ L^{out}_{c,s,k} = L_{c,j,s,k} \\ V^{out}_{c,s,k} = V_{c,j,s,k} \\ L_{c,j,s,k} = LIQ_{j,s,k} \cdot x_{c,j,s,k} \\ V_{c,j,s,k} = VAP_{j,s,k} \cdot y_{c,j,s,k} \\ \sum_{c\in\mathsf{C}} \left(V^{in}_{c,s,k} \cdot hV^{in}_{c,s,k} + L^{in}_{c,s,k} \cdot hL^{in}_{c,s,k} \right) = \sum_{c\in\mathsf{C}} \left(V_{c,j,s,k} \cdot hV_{c,j,s,k} + L_{c,j,s,k} \cdot hL_{c,j,s,k} \right) \\ hL_{c,j,s,k} = \Phi^{h}_{c}\left(T_{j,s,k}\right) \\ hV_{c,j,s,k} = \Phi^{H}_{c}\left(T_{j,s,k}\right) \\ hL^{out}_{c,s,k} = hL_{c,j,s,k} \\ hV^{out}_{c,s,k} = hV_{c,j,s,k} \\ \sum_{c\in\mathsf{C}} x_{c,j,s,k} = 1 \\ \sum_{c\in\mathsf{C}} y_{c,j,s,k} = 1 \\ f^{L}_{c,j,s,k} = f^{V}_{c,j,s,k} \\ f^{L}_{c,j,s,k} = \Phi^{fL}_{c}\left(\mathbf{x}_{j,s,k}, T_{j,s,k}, P_{j,s,k}\right) \\ f^{V}_{c,j,s,k} = \Phi^{fV}_{c}\left(\mathbf{y}_{j,s,k}, T_{j,s,k}, P_{j,s,k}\right) \\ c_{s,k} = \Phi^{c1}\left(DC\right) \end{bmatrix} \vee \begin{bmatrix} \neg Z_{s,k} \\ \wedge \\ L^{in}_{c,s,k} = 0 \\ V^{in}_{c,s,k} = 0 \\ L^{out}_{c,s,k} = 0 \\ V^{out}_{c,s,k} = 0 \\ L_{c,j,s,k} = 0 \\ V_{c,j,s,k} = 0 \\ hL_{c,j,s,k} = 0 \\ hV_{c,j,s,k} = 0 \\ hL^{in}_{c,s,k} = 0 \\ hV^{in}_{c,s,k} = 0 \\ hL^{out}_{c,s,k} = 0 \\ hV^{out}_{c,s,k} = 0 \\ c_{s,k} = 0 \end{bmatrix}$$

$$c \in \mathsf{C}, j \in \mathsf{J1}, s \in \mathsf{S}, k \in \{k \in \mathsf{K} \mid k = 3\} \qquad (5.8)$$

$$\begin{bmatrix} Z^{L}_{s,k} \\ \wedge \\ tL^{in}_{c,s,k} = tL^{out}_{c,s,k} \\ htL^{in}_{c,s,k} = htL^{out}_{c,s,k} \end{bmatrix} \vee \begin{bmatrix} \neg Z^{L}_{s,k} \\ \wedge \\ tL^{in}_{c,s,k} = 0 \\ tL^{out}_{c,s,k} = 0 \\ htL^{in}_{c,s,k} = 0 \\ htL^{out}_{c,s,k} = 0 \end{bmatrix} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \mathsf{K} \qquad (5.9)$$

$$\begin{bmatrix} Z^{V}_{s,k} \\ \wedge \\ tV^{in}_{c,s,k} = tV^{out}_{c,s,k} \\ htV^{in}_{c,s,k} = htV^{out}_{c,s,k} \end{bmatrix} \vee \begin{bmatrix} \neg Z^{V}_{s,k} \\ \wedge \\ tV^{in}_{c,s,k} = 0 \\ tV^{out}_{c,s,k} = 0 \\ htV^{in}_{c,s,k} = 0 \\ htV^{out}_{c,s,k} = 0 \end{bmatrix} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \mathsf{K} \qquad (5.10)$$

By definition, not any pure logical restriction is applied in BGR. Conditional units may exist simultaneously, according to BGR because it is automatically generated from the supergraph.

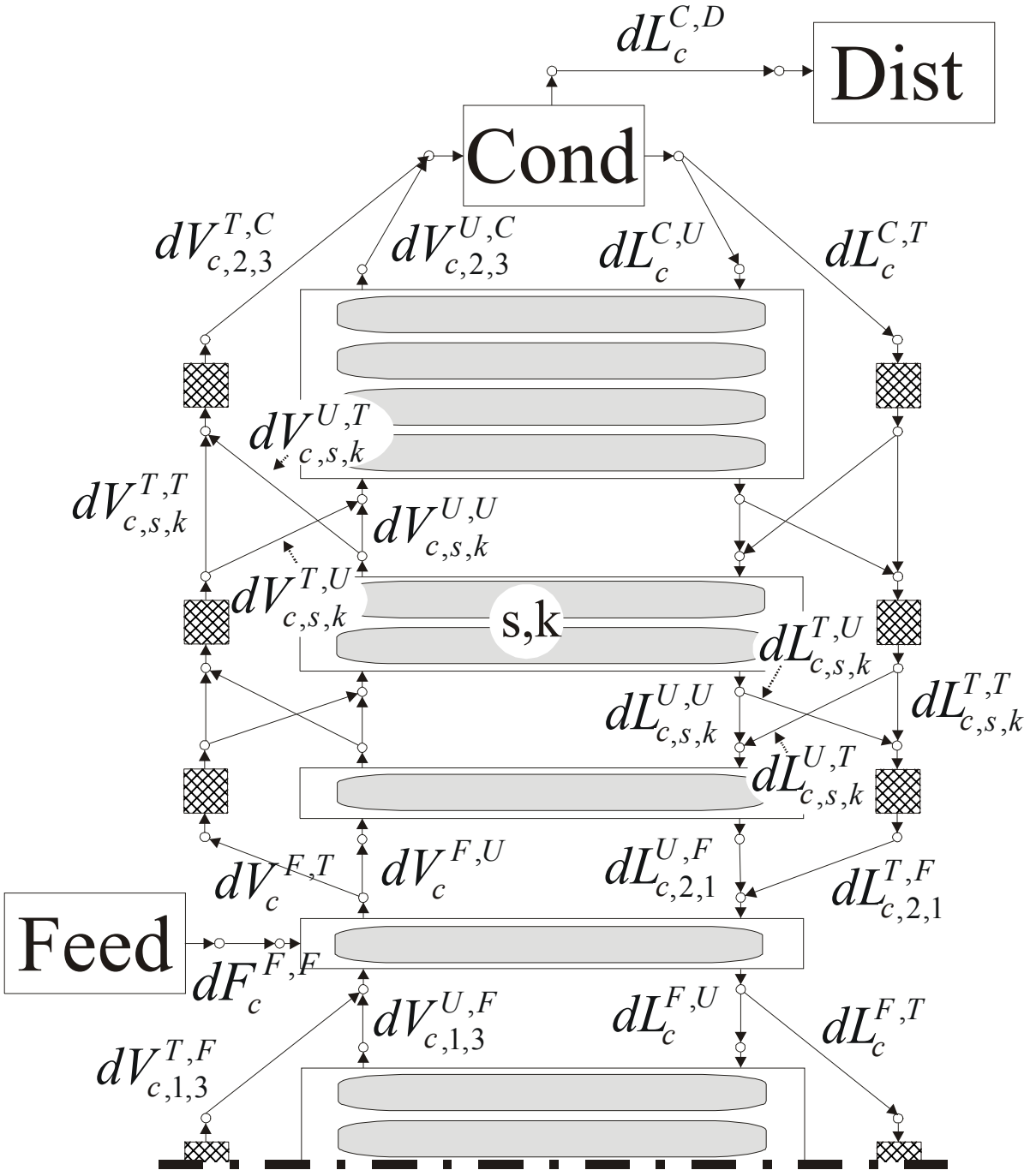


Figure 5.2a. Component material flowrate variables of the edges

Two variables are attached to each stream between units: component flowrates (*dL* or *dV*) and component molar enthalpies (*hdL* or *hdV*). These variables are shown in Figs. 5.2. The first superscript of these variables denotes the type of the unit from which the edge starts, the

second superscript denotes the type of the unit at which the edge ends (*U*=unit containing equilibrium stages; *T*=transport unit; *F*=feed stage or feed source unit; *C*=condenser; *R*=reboiler; *D*=distillate sink unit; *B*=bottom product sink unit). These variables either have three subscripts or merely one, depending on if the edge starts from an output port of a conditional unit or from a permanent one, respectively. The only or the first subscript denotes the component ($c \in \mathsf{C}$). The second subscript (if exists) denotes the column section ($s \in \mathsf{S}$), and the third one denotes the ordinal number of the unit, where the stream comes from, in the column section ($k \in \mathsf{K}$).

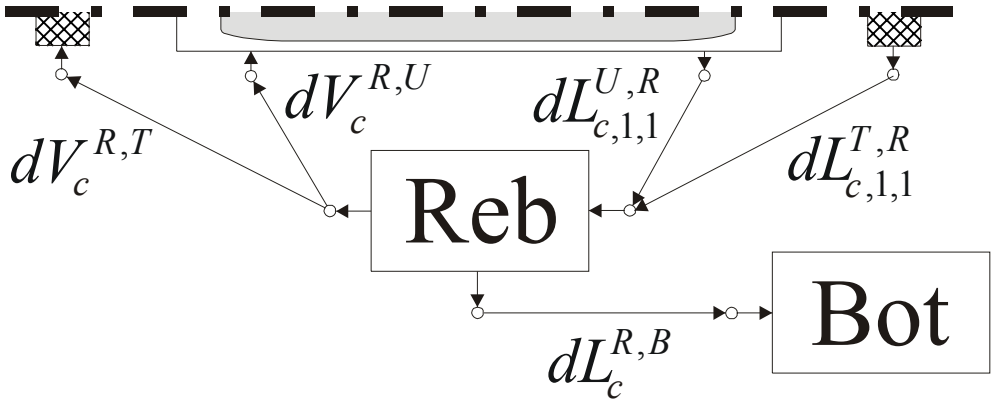


Figure 5.2b. Component material flowrate variables of the edges

The material balances of the input and output ports are expressed by summing the inlet and outlet flowrate variables. For example, the material balances of the input and output vapor ports of the k^{th} unit containing equilibrium stages in the s^{th} column section are as follow:

$$dV_{c,s,k-1}^{U,U} + dV_{c,s,k-1}^{T,U} = V_{c,s,k}^{in} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k \neq \text{first}\} \qquad (5.11)$$

$$V_{c,s,k}^{out} = dV_{c,s,k}^{U,U} + dV_{c,s,k}^{U,T} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k \neq \text{last}\} \qquad (5.12)$$

Similarly, Eq. 5.13 and Eq. 5.14 express the material balances of the input and output port, respectively, of the k^{th} vapor transport unit in the s^{th} column section.

$$dV_{c,s,k-1}^{U,T} + dV_{c,s,k-1}^{T,T} = tV_{c,s,k}^{in} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k \neq \text{first}\} \qquad (5.13)$$

$$tV_{c,s,k}^{out} = dV_{c,s,k}^{T,U} + dV_{c,s,k}^{T,T} \qquad c \in \mathsf{C}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k \neq \text{last}\} \qquad (5.14)$$

The enthalpy balances of the output ports are simple equalities, as it can be seen for the output vapor port of the k^{th} unit containing equilibrium stages in the s^{th} column section (Eqs. 5.15-5.16) and for the output port of the vapor transport unit next to it (Eqs. 5.17-5.18).

$$hdV^{U,U}_{c,s,k} = hV^{out}_{c,s,k} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \qquad (5.15)$$

$$hdV^{U,T}_{c,s,k} = hV^{out}_{c,s,k} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \qquad (5.16)$$

$$hdV^{T,U}_{c,s,k} = htV^{out}_{c,s,k} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{last}\} \qquad (5.17)$$

$$hdV^{T,T}_{c,s,k} = htV^{out}_{c,s,k} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{last}\} \qquad (5.18)$$

The molar enthalpy of the inlet streams, in the input ports, of a unit is calculated from the molar enthalpy of the streams ending in the input port:

$$hV^{in}_{c,s,k} = \Phi^{hdV}_{c}\left(hdV^{U,U}_{c,s,k-1}, hdV^{T,U}_{c,s,k-1}\right) \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \qquad (5.19)$$

$$htV^{in}_{c,s,k} = \Phi^{hdV}_{c}\left(hdV^{U,T}_{c,s,k-1}, hdV^{T,T}_{c,s,k-1}\right) \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \qquad (5.20)$$

These equations may be highly non-convex. In the MINLP representation discussed later, however, a unit containing equilibrium stages and the corresponding transport units cannot exist simultaneously. In that case, the molar enthalpy of the inlet stream of a unit can be simply calculated as the sum of the molar enthalpies carried along the edges from the previous transport unit and the previous unit containing equilibrium stages. Such a technique can be applied because only one of them can be positive, the other one is constrained to be zero.

The material and enthalpy balances of all the input and output ports can be generated similarly.

The objective function is given in our model as the total cost of the column. Since the variable cost is calculated in the condenser and the reboiler (c^C and c^R), and the fix cost related to the units is calculated in the unit relations of each unit, the total cost is simply expressed as the sum of these cost parts:

$$C = c^R + c^C + c^F + \sum_{s \in \mathbf{S}} \sum_{k \in \mathbf{K}} c_{s,k} \qquad (5.21)$$

Logical relations are used in the unit relations, merely to express the conditionality of the conditional units. No other logical constraints are applied in the BGR; hence, it represents all the subgraphs of the R-supergraph.

5.3. Basic MINLP Representation

The Basic MINLP Representation (BMR) can be created automatically from the Basic GDP Representation (GDP). The logical relations are transformed to algebraic equations, and the logical variables (Z) are substituted by binary ones (z). When transforming the BGR to BMR, the equations of the permanent units, the material balances of the input and output ports, and the objective function, remain unchanged. In contrary, some relations of the conditional units are converted to equations including binary variables by certain transforming techniques (e.g. Big M, Multi M, Convex hull formulations). The BMR does not include any additional constraint; thus, it represents all the subgraphs of the R-supergraph, just like the original GDP formulation represents them.

The detailed BMR of our problem is not presented here because it has only theoretical advantage, as a reference representation during a comparison between MR-s. BMR is used in the model generation phase only, but not in the optimisation, because it contains several redundant and unnecessary equations.

5.4. MINLP Representation

The BMR derived automatically from the BGR can be simplified by applying some modifications, in order to decrease the computational difficulties occurring in the course of solution. Experienced engineers can generate an MR directly from BGR. This step is shown in this section.

The logical relations are transformed to algebraic equations, using Big M technique. This transformation is shown here on the example of the third unit, containing four equilibrium stages, in a column section. When the unit exists in a structure then the unit equations have to be satisfied, and if it does not exist then the component flowrate, enthalpy, and cost variables, take zero value, according to the unit relations of the third unit in BGR (Eq. 5.7).

This behaviour of the flowrate variables can be expressed by taking over the material balances from BGR, and introducing additional equations enforcing input flowrate variables to take zero value when the actual unit does not exist. Not only the input flowrate variables, but all the inner and output molar flowrate variables take zero value in this way, as a consequence of the balances, when the unit does not exist. Eqs. 5.22-5.26 list the material balances taken over from BGR, and Eq. 5.27 is the applied Big M equation forcing the input molar flowrate variables to take zero value when the unit does not exist. The binary variable of the unit ($z_{s,k}$)

expresses the existence of the unit; i.e., its value is zero when the unit does not exist. When the unit exist, i.e. the binary variable is unity, the input molar liquid and vapor variables can take positive value; M^{LV} is the upper bound of the right hand side expression in Eq. 5.27. The input liquid and vapor flowrate variables appear in the same Big M constraint in order to use as few equations containing binary variables as possible.

$$V^{in}_{c,s,k} + L_{c,j+1,s,k} = V_{c,j,s,k} + L_{c,j,s,k} \qquad c \in \mathsf{C}, j \in \mathsf{J4}^{first}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k = 3\} \qquad (5.22)$$

$$V_{c,j-1,s,k} + L_{c,j+1,s,k} = V_{c,j,s,k} + L_{c,j,s,k} \qquad c \in \mathsf{C}, j \in \mathsf{J4}^{inner}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k = 3\} \qquad (5.23)$$

$$V_{c,j-1,s,k} + L^{in}_{c,s,k} = V_{c,j,s,k} + L_{c,j,s,k} \qquad c \in \mathsf{C}, j \in \mathsf{J4}^{last}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k = 3\} \qquad (5.24)$$

$$L^{out}_{c,s,k} = L_{c,j,s,k} \qquad c \in \mathsf{C}, j \in \mathsf{J4}^{first}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k = 3\} \qquad (5.25)$$

$$V^{out}_{c,s,k} = V_{c,j,s,k} \qquad c \in \mathsf{C}, j \in \mathsf{J4}^{last}, s \in \mathsf{S}, k \in \{k \mid k \in \mathsf{K}, k = 3\} \qquad (5.26)$$

$$\sum_{c \in \mathsf{C}} \left(L^{in}_{c,s,k} + V^{in}_{c,s,k} \right) \leq M^{LV} \cdot z_{s,k} \qquad s \in \mathsf{S}, k \in \mathsf{K} \qquad (5.27)$$

The molar enthalpy variables can be managed similarly. The enthalpy balance can be taken invariantly from BGR (Eqs. 5.28-5.32), but an additional Big M constraint is necessary to force the input molar enthalpy variables to take zero value when the unit does not exist. As a consequence of the form of Eqs. 5.28-5.30, however, the flowrates may take zero value when the unit does not exist, because of Eq. 5.27, and the enthalpy balances may be satisfied with non-zero molar enthalpies of the streams around the equilibrium stages. If it happens then the output stream enthalpy variables have positive value because of Eqs. 5.31-5.32. In such a case the streams from non-existing unit have positive enthalpy, involving problems in further calculations. To prevent occurring this problem, not only the input, but the output stream enthalpy variables also appear in the Big M constraints applied to them (Eq. 5.33). As a result, all the input and output stream enthalpy variables of the unit (not of the equilibrium stages inside the unit) are set to zero, when the unit does not exist. The enthalpy variables of the equilibrium stages in the unit may take positive value in this case, but it does not involve any problem in the calculations.

$$\sum_{c\in\mathbf{C}}\left(V^{in}_{c,s,k}\cdot hV^{in}_{c,s,k}+L_{c,j+1,s,k}\cdot hL_{c,j+1,s,k}\right)=\sum_{c\in\mathbf{C}}\left(V_{c,j,s,k}\cdot hV_{c,j,s,k}+L_{c,j,s,k}\cdot hL_{c,j,s,k}\right)$$
$$j\in\mathbf{J4}^{first},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.28)$$

$$\sum_{c\in\mathbf{C}}\left(V_{c,j-1,s,k}\cdot hV_{c,j-1,s,k}+L_{c,j+1,s,k}\cdot hL_{c,j+1,s,k}\right)=\sum_{c\in\mathbf{C}}\left(V_{c,j,s,k}\cdot hV_{c,j,s,k}+L_{c,j,s,k}\cdot hL_{c,j,s,k}\right)$$
$$j\in\mathbf{J4}^{inner},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.29)$$

$$\sum_{c\in\mathbf{C}}\left(V_{c,j-1,s,k}\cdot hV_{c,j-1,s,k}+L^{in}_{c,s,k}\cdot hL^{in}_{c,s,k}\right)=\sum_{c\in\mathbf{C}}\left(V_{c,j,s,k}\cdot hV_{c,j,s,k}+L_{c,j,s,k}\cdot hL_{c,j,s,k}\right)$$
$$j\in\mathbf{J4}^{last},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.30)$$

$$hL^{out}_{c,s,k}=hL_{c,j,s,k} \qquad c\in\mathbf{C},j\in\mathbf{J4}^{first},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.31)$$

$$hV^{out}_{c,s,k}=hV_{c,j,s,k} \qquad c\in\mathbf{C},j\in\mathbf{J4}^{last},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.32)$$

$$\sum_{c\in\mathbf{C}}\left(hL^{in}_{c,s,k}+hV^{in}_{c,s,k}+hL^{out}_{c,s,k}+hV^{out}_{c,s,k}\right)\le 4\cdot n_c\cdot M^h\cdot z_{s,k} \qquad s\in\mathbf{S},k\in\mathbf{K} \qquad (5.33)$$

The concentration, fugacity, temperature and pressure variables of the equilibrium stages have to satisfy the unit relations when the unit exist, but there is not any constraint to them when the unit does not exist. These variables can also be constrained to take zero value when the unit does not exist. But such constraints could be too strict, and would involve problems in the MINLP optimisation to find the optimal solution. Therefore, only those constraints are given that enforce the equations in BGR to be satisfied when the unit exists.

This could be achieved with Big M formula by inserting two inequalities to each variable. Such a pair of equations would be for the equation expressing the molar liquid enthalpy in function of the temperature, for example, as follow:

$$-M^h\cdot\left(1-z_{s,k}\right)\le hL_{c,j,s,k}-\Phi^h_c\left(T_{j,s,k}\right) \qquad c\in\mathbf{C},j\in\mathbf{J4},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.34)$$

$$hL_{c,j,s,k}-\Phi^h_c\left(T_{j,s,k}\right)\le M^h\cdot\left(1-z_{s,k}\right) \qquad c\in\mathbf{C},j\in\mathbf{J4},s\in\mathbf{S},k\in\{k\mid k\in\mathbf{K},k=3\} \qquad (5.35)$$

The binary variable ($z_{s,k}$) would take unity, and the left hand side of Eq. 5.34 and the right hand side of Eq. 5.35 would take zero when the unit exists; therefore, the molar liquid enthalpy would become equal to the function calculated from the temperature.

But, in order to decrease the number of equations containing binary variables, another method is suggested. The equation reduced to one side is taken equal to the difference of two non-negative continuous variables Δ (Eq. 5.36). For these two variables Δ, a Big M constraint, including the binary variable of the unit, is added (Eq. 5.37). These Δ variables take zero value, and the difference of two zero values in Eq. 5.37 will take zero value when the unit

exists; therefore, the molar liquid enthalpy has to be equal to the function calculated from the temperature. When the unit does not exist, these Δ variables can take any non-negative value below their upper limit. The difference of two non-negative values in the right hand side of Eq. 5.36 can take positive, or even negative, value in this case; therefore, the original equation need not be satisfied, and the liquid enthalpy variable can take greater or lower value than that calculated from the temperature .

$$hL_{c,j,s,k} - \Phi_c^h\left(T_{j,s,k}\right) = \Delta_{s,k}^{h,+} - \Delta_{s,k}^{h,-} \qquad c \in \mathsf{C}, j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.36)$$

$$\Delta_{s,k}^{h,+} + \Delta_{s,k}^{h,-} \le M^k \cdot \left(1 - z_{s,k}\right) \qquad s \in \mathsf{S}, k \in \mathsf{K} \qquad (5.37)$$

Eq. 5.34-5.35 containing a binary variable are transformed to Eq. 5.36-5.37 with introducing two continuous variables (Δ-s). As a result, only one of the two new equations contain the binary variable, namely Eq. 5.37. This modification resulted in shorter runtime.

The same Δ variables are used also for the equations containing vapor enthalpy variables (Eq. 5.38). The relaxation of the equations will be a bit worse in this case because the upper bounds are not fitted for different variables, but the number of equations containing binary variables are further decreased:

$$hV_{c,j,s,k} - \Phi_c^H\left(T_{j,s,k}\right) = \Delta_{s,k}^{h,+} - \Delta_{s,k}^{h,-} \qquad c \in \mathsf{C}, j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.38)$$

Analogous Big M equations are written for the other equations containing logical expression(s) in the BGR:

$$\sum_{c \in \mathsf{C}} x_{c,j,s,k} - 1 = \Delta_{s,k}^{x,+} - \Delta_{s,k}^{x,-} \qquad j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.39)$$

$$\sum_{c \in \mathsf{C}} y_{c,j,s,k} - 1 = \Delta_{s,k}^{x,+} - \Delta_{s,k}^{x,-} \qquad j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.40)$$

$$\Delta_{s,k}^{x,+} + \Delta_{s,k}^{x,-} \le M^x \cdot \left(1 - z_{s,k}\right) \qquad s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.41)$$

$$f_{c,j,s,k}^L - f_{c,j,s,k}^V = \Delta_{s,k}^{f,+} - \Delta_{s,k}^{f,-} \qquad c \in \mathsf{C}, j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.42)$$

$$f_{c,j,s,k}^L - \Phi_c^{fL}\left(\mathbf{x}_{j,s,k}, T_{j,s,k}, P_{j,s,k}\right) = \Delta_{s,k}^{f,+} - \Delta_{s,k}^{f,-} \qquad c \in \mathsf{C}, j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.43)$$

$$f_{c,j,s,k}^V - \Phi_c^{fV}\left(\mathbf{y}_{j,s,k}, T_{j,s,k}, P_{j,s,k}\right) = \Delta_{s,k}^{f,+} - \Delta_{s,k}^{f,-} \qquad c \in \mathsf{C}, j \in \mathsf{J4}, s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.44)$$

$$\Delta_{s,k}^{f,+} + \Delta_{s,k}^{f,-} \le M^f \cdot \left(1 - z_{s,k}\right) \qquad s \in \mathsf{S}, k \in \left\{k \mid k \in \mathsf{K}, k = 3\right\} \qquad (5.45)$$

Each component flow rate (liquid or vapor) is expressed as the product of the total flowrate ($LIQ_{j,s,k}$ or $VAP_{j,s,k}$) in the equilibrium stages and the mole fraction (Eqs. 5.46-5.47). These equations are applied to calculate the total flow rates $LIQ_{j,s,k}$ and $VAP_{j,s,k}$:

$$L_{c,j,s,k} = LIQ_{j,s,k} \cdot x_{c,j,s,k} \qquad c \in \mathbf{C}, j \in \mathbf{J4}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k = 3\} \qquad (5.46)$$

$$V_{c,j,s,k} = VAP_{j,s,k} \cdot y_{c,j,s,k} \qquad c \in \mathbf{C}, j \in \mathbf{J4}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k = 3\} \qquad (5.47)$$

The cost related to the unit is calculated when the unit exists, and takes zero value when the unit does not exist, according to the unit relations in BGR. This can be expressed with two Δ variables and a Big M constraint similarly to Eqs. 5.36-5.45:

$$c_{s,k} - \Phi^{c3}(DC) = \Delta_{s,k}^{c3,+} - \Delta_{s,k}^{c3,-} \qquad s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k = 3\} \qquad (5.48)$$

$$\Delta_{s,k}^{c,+} + \Delta_{s,k}^{c,-} \le M^{c} \cdot (1 - z_{s,k}) \qquad s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k = 3\} \qquad (5.49)$$

According to the formulas (Eqs. 5.22-5.33 and Eqs. 5.36-5.49), all the unit relations are satisfied when the equilibrium unit exists, and its inlet and outlet streams are of zero flowrate and zero molar enthalpy when the unit does not exist.

All the conditional unit relations containing equilibrium stages, not only those of the 3rd unit in the above example, are transformed to algebraic equations in the same way.

The unit relations of the transport units are transformed to algebraic equations applying the same principles as above. That is, the material and enthalpy balances are taken over, and Big M constraints are added for the input variables, as follow.

The equations of the liquid transport units are listed below:

$$tL_{c,s,k}^{in} = tL_{c,s,k}^{out} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.50)$$

$$htL_{c,s,k}^{in} = htL_{c,s,k}^{out} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.51)$$

$$\sum_{c \in \mathbf{C}} tL_{c,s,k}^{in} \le n_c \cdot M^{L} \cdot z_{s,k}^{L} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.52)$$

$$\sum_{c \in \mathbf{C}} htL_{c,s,k}^{in} \le n_c \cdot M^{h} \cdot z_{s,k}^{L} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.53)$$

The equations of the vapor transport units are listed below:

$$tV_{c,s,k}^{in} = tV_{c,s,k}^{out} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.54)$$

$$htV_{c,s,k}^{in} = htV_{c,s,k}^{out} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.55)$$

$$\sum_{c \in \mathbf{C}} tV_{c,s,k}^{in} \le n_c \cdot M^{V} \cdot z_{s,k}^{V} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.56)$$

$$\sum_{c \in \mathbf{C}} htV_{c,s,k}^{in} \le n_c \cdot M^{h} \cdot z_{s,k}^{V} \qquad s \in \mathbf{S}, k \in \mathbf{K} \qquad (5.57)$$

The material and enthalpy balances have to be satisfied even when the unit does not exist (Eqs. 5.50-5.51 and Eqs. 5.54-5.55). The Big M constraints are needed to write only for the input variables (Eqs. 5.52-5.53 and Eqs. 5.56-5.57) because the output variables take zero when the input variables are zero, due to the form of the balances.

The above representation is constructed almost automatically from the GDP, but a small enhancement is applied by omitting redundant equations, and it does not include any constraint additional to those specified in the GDP representation; thus, it represents all the subgraphs of the R-supergraph. However, the numerical behaviour of the MINLP representation can be greatly enhanced with some modifications outlined below.
For this aim, some specific properties of the actual chemical engineering problem are taken into account to further improve the numerical behaviour of the MINLP, by tailoring its actual form.

First, the equations used for the calculation of the fix cost of the conditional units are strongly nonlinear. Calculation of the cost, depending on the number of trays, for each conditional unit separately consumes a great effort of computation. Combining these cost functions into a single equation is much better idea. Accordingly, all the parts are calculated in the objective function, instead of summing up the earlier calculated cost parts, as in Eq. 5.21:

$$C = \Phi^{cC}(QC) + \Phi^{cR}(QR) + \Phi^{fix}(N_{st}, DC), \tag{5.58}$$

where N_{st} is the number of the equilibrium stages in the column.

Second, the number of variables can be decreased by omitting the variables belonging to the edges of the R-graph. According to Fig. 5.3, a material balance can be, and are, written around the dashed line:

$$V^{out}_{c,s,k-1} + tV^{out}_{c,s,k-1} = V^{in}_{c,s,k} + tV^{in}_{c,s,k} \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \tag{5.59}$$

Similarly, the enthalpy calculations can be, and are, written in a simpler form, as follow:

$$hV^{in}_{c,s,k} = \Phi^{hdV}_{c}\left(hV^{out}_{c,s,k-1}, htV^{out}_{c,s,k-1}\right) \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \tag{5.60}$$

$$htV^{in}_{c,s,k} = \Phi^{hdV}_{c}\left(hV^{out}_{c,s,k-1}, htV^{out}_{c,s,k-1}\right) \qquad c \in \mathbf{C}, s \in \mathbf{S}, k \in \{k \mid k \in \mathbf{K}, k \neq \text{first}\} \tag{5.61}$$

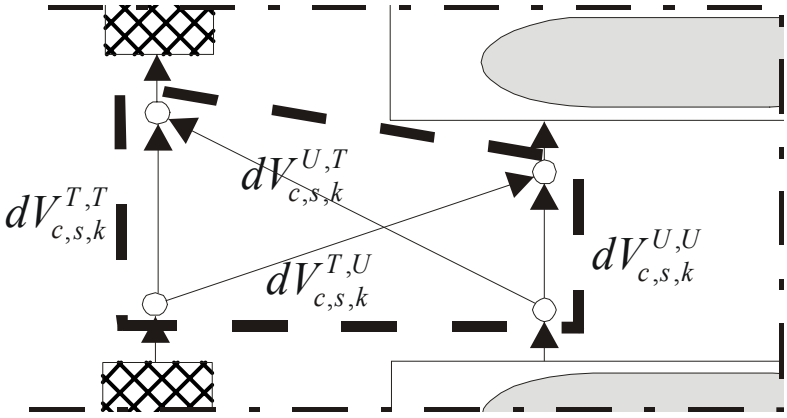


Figure 5.3. Omitting the edge variables

All the edge variables (*dL*, *dV*, *hdL*, *hdV*) are omitted in this way, and even the number of equations related to the input and output ports (Eqs. 5.11-5.20) is decreased.

5.5. Ideal MINLP Representation

As mentioned earlier, neither the BGR nor the MR described in section 5.4 includes any logical relation or algebraic equation to exclude the representation of the non-considered structures; therefore, they represent all the subgraphs of the supergraph, i.e. all the feasible structures, not only the considered ones. Accordingly, they represent several non-considered structures, as well. The representation of the non-considered structures and their representing graphs should be excluded from the MINLP representation in order to decrease the chance for incorporating structural redundancy and multiplicity. A representation is called Ideal MINLP Representation (IMR) if it represents the considered graphs and considered structures only.

In the particular case of modelling distillation columns with R-graphs, a unit containing equilibrium stages and the two transport units belonging to it do not exist simultaneously in a considered structure. This constraint can be expressed by a pair of 'exclusive or' (XOR) relations between the logical variables in the following form:

$$Z^U_{s,k} \oplus Z^L_{s,k}. \qquad s \in \mathsf{S}, k \in \mathsf{K} \tag{5.62}$$

$$Z^U_{s,k} \oplus Z^V_{s,k}. \qquad s \in \mathsf{S}, k \in \mathsf{K} \tag{5.63}$$

These logical relations can be expressed by algebraic equation as well (see Raman and Grossmann, 1991):

$$z_{s,k} + z^{L}_{s,k} = 1 \qquad s \in \mathbf{S}, k \in \mathbf{K} \tag{5.64}$$

$$z_{s,k} + z^{V}_{s,k} = 1 \qquad s \in \mathbf{S}, k \in \mathbf{K} \tag{5.65}$$

The MR described in section 5.4 is made ideal by attaching Eqs. 5.64-5.65 to it. The new (ideal) MINLP representation formed in this way represents the considered structures only.

5.6. Binarily Minimal MINLP Representation

The computational difficulties occurring during the solution of an MILP / MINLP problem usually increase with the number of binary variables. Therefore, using as few number of binary variables as possible is a good idea. This can be achieved by modifying the MINLP representation; but in some cases, like the one here, it can also be achieved by modifying the superstructure, and the supergraph. A well shaped supergraph is instructive in forming effective MINLP representation.

As it was mentioned above, an array of n_{bv} binary variables, applied to distinguish structures, can take $2^{n_{bv}}$ different values. This $2^{n_{bv}}$ has to be at least as large as the number of different graphs. If the supergraph includes n_g subgraphs, then the minimal number of the binary variables needed for making distinction between them is the lowest integer number which satisfies Eq. 5.66.

$$n_{bv} \geq \log_2 n_g \tag{5.66}$$

An MINLP representation applying minimum number of binary variables for distinguishing different structures is called Binarily Minimal MINLP Representation (BMMR).

Consider a distillation column including, for example, 15 equilibrium stages (7 stages above, 7 stages below, and the feed stage), shown in Fig. 5.1. In the column section between the reboiler and the feed stage, selected for an example, 8 different structure variations are possible according to the number of equilibrium stages: 0, 1, 2,…, 7 equilibrium stages, in this particular case. It follows that $3=\log_2 8$ binary variables are necessary to describe this column section for BMMR.

The number of equilibrium stages in the k^{th} unit (counted from bottom up) is 2^{k-1} in the superstructure. All the possible stage numbers can be generated with this configuration. For example, 3 stages can be assembled using the first and the second unit (Fig. 5.4a), 4 stages are

assigned by using the third unit (Fig. 5.4b), 5 stages are assigned by using the first and the third units (Fig. 5.4c), and so on. When a unit is not used in a structure, then it is by-passed through the transport units.

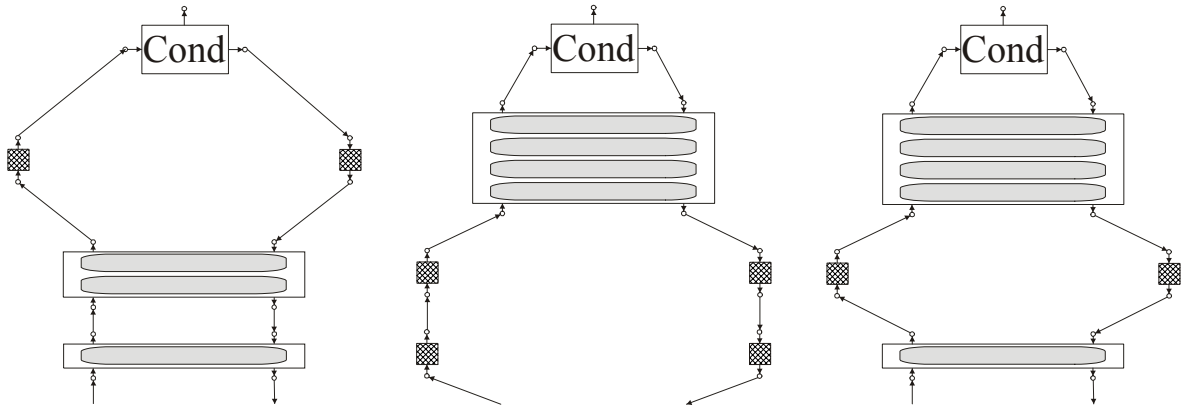


Figure 5.4. Column section containing different number of equilibrium stages

There are nine binary variables in our example column section: 3 for the units containing equilibrium stages, 3 for the liquid transport units, and 3 for the vapor transport units. If the number of binary variables were decreased to 3, then the minimum would be reached, and the BMMR would be generated. How this decrease has been achieved is shown below.

In section 5.5, the logical relations between logical variables were rewritten in algebraic form between the binary variables of the units containing equilibrium stages and the binary variables of the transport units (Eqs. 5.64-5.65). The binary variables in these equations are now substituted with the following expressions:

$$z^L_{s,k} := 1 - z_{s,k} \qquad s \in \mathbf{S}, k \in \mathbf{K} \tag{5.67}$$

$$z^V_{s,k} := 1 - z_{s,k} \qquad s \in \mathbf{S}, k \in \mathbf{K} \tag{5.68}$$

If the maximal number of equilibrium stages is increased, the number of the conditional units and, in turn, the number of binary variables have also to be increased. If the maximum number of equilibrium units is increased up to, e.g., 8 in a column section then, according to Eq. 5.66, $\log_2 9 \approx 3.17 \leq 4$ binary variables have to be used in BMMR. If a fourth unit containing $2^{4-1}=8$ equilibrium stages is added to the superstructure, then the BMMR can be generated in the same way as it was done above. Of course, with this fourth unit, the maximum number of the units is increased from 8 to 15; therefore, additional constraints are

needed to exclude the representation of the non-considered structures (containing 10 to15 equilibrium stages in a column section). Such a unit containing 8 equilibrium stages can be inserted automatically, even in the case of maximum 15 equilibrium stages in a column section.

The MINLP Representation obtained through the above detailed methods uses minimal number of binary variables to distinguish the structures; therefore, it is a Binarily Minimal MINLP Representation. Nevertheless, it still preserves its ideality, thus it is Binarily Minimal and Ideal MINLP Representation (BMIMR), as well.

5.7. Examples

The BMIMR developed according to section 5.6 is tested from computational point of view on three different separation examples. The new model was intended to compare with the GDP model of Yeomans and Grossmann (2000a). Since GDP solver was not available for us, that GDP model was transformed to MINLP model using Big M technique, and the results obtained with that MINLP model was compared to the results obtained with our BMIMR model

All the examples were solved on a Sun Sparc Station using GAMS (Brooke et al., 1992). The MINLP solver was DICOPT++ (Viswanathan and Grossmann, 1990). The NLP subproblems were solved with CONOPT2; the MILP subproblems were solved with CPLEX.

In all the examples the cost function of Luyben and Floudas (1994) were applied:

$$C = \frac{\beta_{tax}(c_{LPS} \cdot QR + c_{CW} \cdot QC)V + UF \cdot \Phi^{fix}(N_{st}, DC) / \beta_{pay}}{1000} \tag{5.69}$$

$$\Phi^{fix}(N_{st}, DC) = 12.3[615 + 324DC^2 + 486(6 + 0.76N_{st})DC] + 245N_{st}(0.7 + 1.5DC^2) \tag{5.70}$$

where β_{tax} is the tax factor (=0.4); c_{LPS} is the cost of the low pressure steam (=1.1488·10^{-6} USD/kJ); c_{CW} is the cost of the cooling water (=3.73·10^{-8} USD/kJ); UF is the update factor (=1.292); β_{pay} is the payback period (=4 yr).

The column diameter is calculated from cross section of the column (A, [m^2]):

$$DC = 2\sqrt{\frac{A}{\pi}} \tag{5.71}$$

The cross section of the column was determined by the flowrate of the vapor stream and the density of the vapor in the reboiler, using the F_f–procession (Kister, 1995):

$$A = \frac{\dot{m}_V}{F_{\max}\sqrt{\rho_V}} \tag{5.72}$$

where $\dot{m}_V$ is the mass flowrate of the vapor [kg/s]; $F_{\max}$ is the F-factor ($=2.2\sqrt{Pa}$); and ρ_V is the density of the vapor [kg/m^3].

The original objective function was divided by 1000 for better scaling.

5.7.1. Example 5.1

Example 5.1 involves the separation of a benzene-toluene mixture. Equimolar feed is considered, i.e. the charge composition is $\mathbf{x}_{ch}$=[0.5; 0.5]. The feed is 100 kmol/h. The specified purity is 0.98 benzene in the distillate, and 0.98 toluene in the bottom product. Atmospheric column is used, P=760 torr. The mixture is assumed be ideal, and constant molar overflow is also assumed.

The vapor-liquid equilibrium is calculated according to the Raoult-Dalton equation:

$$f_c^V = P \cdot y_c \tag{5.73}$$

$$f_c^L = x_c \cdot p_c^0(T) \tag{5.74}$$

where $p_c^0(T)$ is the vapor pressure of component c. Vapor pressure $p_c^0(T)$ is calculated with Antoine equation:

$$\log\left(p_c^0[\text{Hgmm}]\right) = A_c - \frac{B_c}{C_c + T[^\circ\text{C}]}, \tag{5.75}$$

where A_c, B_c and C_c are the Antoine constants of component c. The applied constants (Gmehling et al., 1977) are collected in Table A2 in Appendix.

The maximum number of equilibrium stages in the column was set to 63 (31 above, 31 below the feed, and the feed stage).

The initial values of the variables were calculated by modelling the process using the maximum number of stages and the specified product purity. The modelling software was ChemCAD. The stopping criterion was the execution of the specified maximum number of iterations.

Table 5.1. Model characteristics of Example 5.1

Model	No. of eqs.	No. of non-lin. eqs.	No. of vars.	No. of bin. vars.
Yeomans-based MINLP	2051	514	1272	60
New	1592	519	1449	10

Table 5.2. Computational results of Example 5.1

Model	N_{st}	*DC*	*Ref*	Objective function	No. of iters.	Solution. time (CPU s)
Yeomans-based MINLP	13	1.38	2.83	83.15	150	79 565
New	16	1.18	1.76	73.12	150	13 643

Data characterizing the models are listed in Table 5.1, as follow: number of equations; number of non-linear equations amongst them; number of variables; and number of binary variables amongst them. The new model uses slightly more variables and less equations than the Yeomans model; but the number of binary variables is decreased significantly, since the new model is a binarily minimal one.

Table 5.2 collects the data of the optimal solution: the number of equilibrium stages; the column diameter [m]; the reflux ratio; and the total cost of the column in [1000 USD/yr]. The last two columns show the number of main iterations, and the computation time needed to find the optimal solution.

The new model found a better solution than the Yeomans-based MINLP model, in 150 iterations; and the solution time was also significantly faster, with almost 83 %.

5.7.2. Example 5.2

In this example, the target is separating equimolar methanol-propanol-buthanol mixture. The feed flow rate is 100 kmol/h. The specified purity is 0.99 methanol in the distillate, 0.99 buthanol in the bottom product. The same assumptions are used as in Example 5.1 (ideal mixture, CMO, constant atmospheric pressure). The phase equilibrium is calculated also with the Raoult-Dalton equation (Eqs. 5.73-5.74), and the vapor pressure with the Antoine equation (Eq. 5.75). The Antoine constants (Gmehling et al., 1977) are listed in Table A3 in Appendix.

The maximum number of stages was set to 63. The initial values were calculated by modelling the process using the maximal number of stages and the specified prescribed

product purity. The modelling software was ChemCAD. The stopping criterion was a specified maximum number of iterations.

The model characteristics are presented in Table 5.3, and the computational results in Table 5.4.

Table 5.3. Model characteristics of Example 5.2

Model	No. of eqs.	No. of non-lin. eqs.	No. of vars.	No. of bin. vars.
Yeomans-based MINLP	2983	515	1778	60
New	2224	520	2027	10

Table 5.4. Computational results of Example 5.2

Model	N_{st}	*DC*	*Ref*	Objective function	No. of iters.	Solution. time (CPU s)
Yeomans-based MINLP	41	0.67	0.825	68.257	150	81 471
New	15	0.71	1.052	43.058	150	90 450

In this case the model characteristics are similar to those in Example 5.1. The new model contains a bit more non-linear equations and variables, but less number of equations, and significantly less number of binary variables. The Yeomans-based MINLP model ran through the 150 iterations about 10 % faster than the new model, but the found optimum of the latter is better with almost 37 %.

5.7.3. Example 5.3

In Example 5.3 equimolar ethanol-water mixture is chosen as feed stream to be separated. The feed flowrate is 100 kmol/h. The required purity is 0.85 ethanol in the distillate and 0.999 water in the bottom product. The CMO and constant atmospheric pressure assumptions were used. The phase equilibrium is calculated with the modified Raoult-Dalton equation:

$$f_c^V = P \cdot y_c \tag{5.76}$$

$$f_c^L = \gamma_c \cdot x_c \cdot p_c^0(T) \tag{5.77}$$

where γ_c is the activity coefficient of component *c* calculated by the Margules-equations:

$$\ln \gamma_1 = [A_{12} + 2 \cdot (A_{21} - A_{12}) \cdot x_1] \cdot x_2^2 \tag{5.78}$$

$$\ln \gamma_2 = [A_{21} + 2 \cdot (A_{12} - A_{21}) \cdot x_2] \cdot x_1^2 \tag{5.79}$$

The vapor pressure is modelled with the Antoine equations (Eq. 5.75). The model parameters (Gmehling et al., 1977) are listed in Tables A4-A5 in Appendix.

The maximum number of stages was set to 63. The initial values of the variables were calculated by modelling the process using the maximum number of stages and the specified product purity. The modelling software was ChemCAD. The stopping criterion was a specified maximum number of iterations.

The model characteristics are collected in Table 5.5, and the computational results in Table 5.6.

Table 5.5. Model characteristics of Example 5.3

Model	No. of eqs.	No. of non-lin. eqs.	No. of vars.	No. of bin. vars.
Yeomans-based MINLP	2177	640	1398	60
New	1718	645	1575	10

Table 5.6. Computational results of Example 5.3

Model	N_{st}	*DC*	*Ref*	Objective function	No. of iters.	Solution. time (CPU s)
Yeomans-based MINLP	47	0.89	2.38	121.9	150	181 564
New	26	0.95	2.87	100.8	150	10 929

The new model found a better solution 94 % faster than the Yeomans-based MINLP model.

For the sake of completeness, it has to be remarked that the models were tested with the SBB MINLP solver, as well. That solver uses the modified branch and bound algorithm, instead of outer approximation. However, we cannot conclude unambiguous issue in this case, because of the very wide scattering of the results.

5.8. Summary

A new, R-graph based, superstructure and MINLP model have been developed for distillation column synthesis. The improvement of the MINLP model is performed not only during the formulation of the mathematical model, but already the superstructure is generated in a shape as to have better MINLP formulation.

First, the Basic GDP Representation is generated automatically, based on the superstructure. An MINLP Representation is formulated by transforming the logical relation into algebraic ones, and omitting the redundant equations and variables.

The MINLP model is idealized excluding the representation of the non-considered graphs and structures. Finally, the number of binary variables is decreased to the minimum. Namely, the Binarily Minimal and Ideal MINLP Representation is generated.

The final MINLP model is compared to the MINLP transformed from the GDP model of Yeomans and Grossmann (2000a). The computational results of three examples show that the solution of the new model needs less computational time, and provides better local optima.

6. Major new results

1. **Case-based reasoning is suitable for selecting a superstructure with an MINLP model of mathematical programming in distillation column synthesis.** A case-based reasoning method has been developed which can retrieve the most similar case from a data base in distillation column synthesis problems of ideal mixture containing maximum five components. First, a set of matching cases is retrieved using inductive retrieval. The cases are classified according to the operational attributes like sharp/non-sharp separation, heat integration, number of products and feeds. Then the cases of the set are ranked using nearest neighbour method according to their similarities to the actual problem considering the component types, the boiling point and molar masses of the components, the feed and the product compositions. After the retrieval, the three most similar cases are reported. The superstructure can be used in the solution of the actual problem. The MINLP model and the solution of the selected cases can be adapted to the actual requirements, and the adopted solution can be used as an initial point during the optimisation.

2. **An automated procedure has been developed for the generation of Basic MINLP Representation (BMR) which can serve as a reference to study whether an MINLP representation represents a supergraph or not.** First, the Basic GDP Representation (BGR) is formulated based on the R-graph representation of the superstructure. BGR contains the constraints of the units in disjunctive form, the balances of input and output ports, and the cost function. BGR does not include any additional logical constraints but the unit relations; therefore, it represents all the subgraphs of the supergraph. Then BMR is generated from BGR using binary variables instead of logical ones, and transforming logical relations into algebraic ones. The generation of BGR and BMR is performed automatically from the R-supergraph; therefore, BMR can serve as a reference in the comparison of an MINLP Representation (MR) whether it represents the supergraph or not. **An MR represents the supergraph if a bijective mapping can be given between a subset of the feasible region of MR and the feasible region of BMR.**

3. **An MINLP Representation (MR) is "good" if it represents only the considered graphs of the supergraphs. In this case, MR is called Ideal MINLP Representation (IMR).** IMR can always be constructed by excluding the representations of the non-considered

graphs. The representation of non-considered graphs can be excluded by extending the MR with algebraic transformations of pure logical constraints. Computational results show that idealization of an MR decreases the solution time, and increases the maximum solvable size of the problem.

4. **An MINLP Representation (MR) is "good" if it uses minimal number of binary variables to make distinction between different structures. In this case, MR is called Binarily Minimal MINLP Representation (BMMR).** This means the use of n_{bv} number of binary variables in case of n_g number of represented graphs, where n_{bv} is the smallest whole number that satisfies $n_{bv} \geq \log_2 n_g$. BMMR can always be generated by introducing new binary variables instead of the old ones, and transforming the equations. Computational results show that decreasing the number of binary variables of an MR decreases the solution time and increases the maximum solvable size of the problem.

5. **A new, R-graph based, superstructure, and corresponding MINLP model, have been developed for the synthesis of a single distillation column.** The superstructure is generated in a way that makes possible an easy generation of the Binarily Minimal and Ideal MINLP Representation. The developed MINLP model is Binarily Minimal and Ideal, i.e. it uses minimum number of binary variables to make distinction between structures, and it represents considered structures only. The new MINLP model finds better local optima in shorter computational time than the MINLP model based on the GDP model of Yeomans and Grossmann (2000a).

Publications

Publications on which the theses are based

Papers in international journals

Tivadar Farkas, Yuri Avramenko, Andzrej Kraslawski, Zoltán Lelkes, and Lars Nyström (2005) Selection of proper MINLP model of distillation column synthesis by case-based reasoning. *Industrial & Engineering Chemistry Research*, accepted for publication.

Tivadar Farkas, Endre Rév, and Zoltán Lelkes (2005) Process flowsheet superstructures: Structural multiplicity and redundancy: Part I: Basic GDP and MINLP representations. *Computers & Chemical Engineering*, *29*(10), 2180-2197.

Tivadar Farkas, Endre Rév, and Zoltán Lelkes (2005) Process flowsheet superstructures: Structural multiplicity and redundancy: Part II: Ideal and binarily minimal MINLP representations. *Computers & Chemical Engineering*, *29*(10), 2198-2214.

Tivadar Farkas, Barbara Czuczai, Endre Rév, Zsolt Fonyó, and Zoltán Lelkes (2005) R-graph-based superstructure and MINLP model for distillation column synthesis. *Industrial & Engineering Chemistry Research*, submitted for publication.

Endre Rév, **Tivadar Farkas**, and Zoltán Lelkes (2005) Process flowsheet structures, *International Journal of Computer Mathematics*, submitted for publication.

Presentations at international conferences

Endre Rév, Zoltán Lelkes, and **Tivadar Farkas** (2002) Structural multiplicity and redundancy in chemical process synthesis with MINLP. *Proceeding of SIMO 2002. Systéme d'Information Modélisation, Optimisation Commande en Génie des Procédés*, 24-25 October, 2002, Toulouse, France

Tivadar Farkas, Yuri Avramenko, Andzrej Kraslawski, Zoltán Lelkes, and Lars Nyström (2003) Selection of MINLP model of distillation column synthesis case-based reasoning. *Computer-Aided Chemical Engineering, 14. European Symposium on Computer Aided Process Engineering – 13. ESCAPE-13*, 1-4 June, 2003, Lappeenranta, Finland. *Elsevier*. 113-118.

Tivadar Farkas, Endre Rév, and Zoltán Lelkes (2004) Structural multiplicity and redundancy in chemical process synthesis with MINLP. *Computer-Aided Chemical Engineering, 15. European Symposium on Computer Aided Process Engineering – 14. ESCAPE-14*, 16-19 May, 2004, Lisbon, Portugal. *Elsevier*. 403-408.

Barbara Czuczai, **Tivadar Farkas**, Zsolt Fonyó, and Zoltán Lelkes (2005) R-graph-based distillation column superstructure and MINLP model. *Computer-Aided Chemical Engineering, 16. European Symposium on Computer Aided Process Engineering – 15. ESCAPE-14*, 29 May – 01 June, 2005, Barcelona, Spain. *Elsevier*. 889-894.

Presentations at Hungarian conferences

Zoltán Lelkes, Endre Rév, and **Tivadar Farkas** (2001) *S*zuperstruktúrák optimális MINLP reprezentációi. *Műszaki Kémiai Napok '01. 24-26 April 2001, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 00 6467 7, 289. (in Hungarian)

Tivadar Farkas, Yuri Avramenko, Andzrej Kraslawski, Zoltán Lelkes, and Lars Nyström (2003) Case-based reasoning aided MINLP optimization of distillation column and sequences. *Műszaki Kémiai Napok '03. 8-10 April 2003, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 7172 99 8, 214-218.

Barbara Czuczai, **Tivadar Farkas**, Zsolt Fonyó, and Zoltán Lelkes (2005) Desztillációs kolonna R-gráf alapú szuperstruktúrája és MINLP modellje, *Műszaki Kémiai Napok '05, 26-28 April 2005, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 9495 71 9, 212-215. (in Hungarian)

Other publications connected to the PhD research work

Papers in international journals

Zsolt Szitkai, **Tivadar Farkas**, Zoltán Lelkes, Endre Rév, Zsolt Fonyó, and Zdravko Kravanja (2005) A fairly linear MINLP model for the synthesis of mass exchange networks, *Industrial & Engineering Chemistry Research*, accepted for publication.

Presentations at international conferences

Zoltán Lelkes, Zsolt Szitkai, **Tivadar Farkas**, Endre Rév, and Zsolt Fonyó (2003) Short-cut design of batch extractive distillation using MINLP. *Computer-Aided Chemical Engineering, 14. European Symposium on Computer Aided Process Engineering – 13.* ESCAPE-13, 1-4 June, 2003, Lappeenranta, Finland. *Elsevier*. 203-208.

Zsolt Szitkai, **Tivadar Farkas**, Zdravko Kravanja, Zoltán Lelkes, Endre Rév, and Zsolt Fonyó (2003) A new MINLP model for mass exchange network synthesis. *Computer-Aided Chemical Engineering, 14. European Symposium on Computer Aided Process Engineering – 13. ESCAPE-13*, 1-4 June, 2003, Lappeenranta, Finland. *Elsevier*. 323-328.

Abdulfattah M. Emhamed, Zoltán Lelkes, Endre Rév, **Tivadar Farkas**, Zsolt Fonyó, and Duncan M. Fraser (2005) New hybrid method for mass exchange network optimisation. *Computer-Aided Chemical Engineering, 16. European Symposium on Computer Aided Process Engineering – 15. ESCAPE-14*, 29 May – 01 June, 2005, Barcelona, Spain. *Elsevier*. 877-882.

Zoltán Lelkes, Endre Rév, **Tivadar Farkas**, Zsolt Fonyó, Tibor Kovács, and Ian Jones (2005) Multicommodity transportation and supply problem with stepwise constant cost function. *Computer-Aided Chemical Engineering, 16. European Symposium on Computer Aided Process Engineering – 15. ESCAPE-14*, 29 May – 01 June, 2005, Barcelona, Spain. *Elsevier*. 1069-1074.

Presentations at Hungarian conferences

Zoltán Lelkes, **Tivadar Farkas**, and Endre Rév (2002) Optimization of Batch Extractive Distillation using MINLP. *Műszaki Kémiai Napok '02. 16-18 April 2002, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 7172 95 5, 112-117.

Zsolt Szitkai, **Tivadar Farkas**, Zdravko Kravanja, Zoltán Lelkes, Endre Rév, and Zsolt Fonyó (2004) A new MINLP model for mass exchange network synthesis, *Műszaki Kémiai Napok '04, 20-22 April 2004, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 9495 37 9, 310-313. (in Hungarian)

Zoltán Lelkes, Endre Rév, **Tivadar Farkas**, Zsolt Fonyó, Tibor Kovács, and Ian Jones (2005) Többtermékes szállítási és elosztási probléma optimalizálása, *Műszaki Kémiai Napok '05, 26-28 April 2005, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 9495 71 9, 208-211.

Abdulfattah M. Emhamed, Zoltán Lelkes, Endre Rév, **Tivadar Farkas**, Zsolt Fonyó, and Duncan M. Fraser (2005) New hybrid method for mass exchange network optimisation, *Műszaki Kémiai Napok '05, 26-28 April 2005, Veszprém, KE Műszaki Kémiai Kutató Intézet*, ISBN 963 9495 71 9, 217.

References

Aggarwal, A.; and Floudas, C. A. (1992) Synthesis of heat integrated nonsharp distillation sequences. *Computers & Chemical Engineering*, *16*(2), 89–108.

Bäck, T.; and Schwefel, S. (1993) An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, *1*, 1.

Balas, E. (1985) Disjunctive programming and a hierarchy of relaxations for discrete optimization problems, *SIAM Journal on Algebraic and Discrete Methods*, *6*(3), 466-486.

Barttfeld, M.; Aguirre, P. A.; and Grossmann, I. E. (2003) Alternative representations and formulations for the economic optimization of multicomponent distillation columns. *Computers & Chemical Engineering*, *27*(3), 363-383.

Biegler, L. T.; Grossmann, I. E.; and Westerberg, A. W. (1997) Systematic methods of chemical process design. Prentice Hall, International Series in the Physical and Chemical Engineering Science.

Biegler, L.T.; and Grossmann, I.E. (2004) Retrospective on optimization. *Computers & Chemical Engineering*, *28*(8), 1169-1192.

Borchers, B.; and Mitchell, J. E. (1994) An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers & Operations Research*, *21*(4), 39-367.

Brendel, M. H.; Friedler, F.; and Fan, L. T. (2000) Combinatorial foundation for logical formulation in process network synthesis. *Computers & Chemical Engineering*, *24*(8), 1859–1864.

Brooke, A.; Kendrick, D.; and Maereus, A. (1992) GAMS, A User's Guide, Release 2.25. Scientific Press, Palo Alto.

Caballero, J. A.; and Grossmann, I. E. (1999) Aggregated model for integrated distillation systems. *Industrial & Engineering Chemistry Research*, *38*, 2330–2344.

Caballero, J. A.; and Grossmann, I. E. (2001) Generalized disjunctive programming model for the optimal synthesis of thermally linked distillation columns. *Industrial & Engineering Chemistry Research*, *40*, 2260–2274.

Comeaux, R. G. (2000) Synthesis of MENs with minimum total cost. *MSc thesis*, Dept. of Process Integration, UMIST, Manchester, England.

Coulson, J. M.; Richardson, J. F.; Backhurst, J. R.; and Harker, J. H. (1985) Chemical Engineering. Pergamon Press.

Daichendt, M. M.; and Grossmann, I.E. (1998) Integration of hierarchical decomposition and mathematical programming for the synthesis of process flowsheets. *Computers & Chemical Engineering*, *22*(1-2), 147-175.

d'Anterroches, L.; and Gani, R. (2005) Group contribution based process flowsheet synthesis, design and modelling. *Fluid Phase Equilibria*, *228-229*, 141-146.

Douglas, J. M. (1985) A hierarchical decision procedure for process synthesis. *AIChE Journal*, *31*(3), 353-362.

Douglas, J. M. (1988) Conceptual design of chemical processes. McGraw-Hill, New York.

Duran, M. A.; and Grossmann, I. E. (1986) A mixed-integer nonlinear programming algorithm for process systems synthesis. *AIChE Journal*, *32*(4), 592-606.

El-Halwagi, M. M. (1997) Pollution prevention through process integration. Academic Press, San Diego, California, USA.

El-Halwagi, M. M.; and Manousiouthakis, V. (1989) Synthesis of mass exchange networks, *AIChE Journal*, *35*(8), 1233-1244.

Farkas, T. (2001) Szuperstruktúrák optimális MINLP reprezentációi. In Study report at conference on scientific research by students (TDK dolgozat) (in Hungarian).

Farkas, T.; Avramenko, Y.; Kraslawski, A.; Lelkes, Z.; and Nyström, L. (2005a) Selection of proper MINLP model of distillation column synthesis by case-based reasoning, *Industrial & Engineering Chemistry Research*, accepted for publication.

Farkas, T.; Rév, E.; and Lelkes, Z. (2005b) Process flowsheet superstructures: Structural multiplicity and redundancy: Part I: Basic GDP and MINLP representations. *Computers & Chemical Engineering*, *29*(10), 2180-2197.

Farkas, T.; Rév, E.; and Lelkes, Z. (2005c) Process flowsheet superstructures: Structural multiplicity and redundancy: Part II: Ideal and binarily minimal MINLP representations. *Computers & Chemical Engineering*, *29*(10), 2198-2214.

Floudas, C.A. (1995) Non-linear and mixed-integer optimisation: Fundamentals and applications. Oxford University Press, New York.

Fonyó, Z.; and Mizsey, P. (1990) A global approach to the synthesis and preliminary design of integrated total flowsheets. *AIChE Annual Meeting*, Chicago.

Fraga, E. S.; and Zilinskas, A. (2003) Evaluation of hybrid optimization methods for the optimal design of heat integrated distillation sequences. *Advances in Engineering Software*, *34*(2), 73-86.

Friedler, F.; Tarján, K.; Huang, W. Y.; and Fan L. T. (1992a) Graph-theoretical approach to process synthesis axioms and theorems. *Chemical Engineering Science*, *47*(8), 1973-1988.

Friedler, F.; Tarján, K.; Huang, W. Y.; and Fan L. T. (1992b) Combinatorial algorithms for process synthesis, *Computers & Chemical Engineering*, *16*, S313-320.

Friedler, F.; Tarján, K.; Huang, W. Y.; and Fan, L. T. (1993) Graph-theoretical approach to process synthesis: polynomial algorithm for maximal structure generation. *Computers & Chemical Engineering*, *17*(9), 929-942.

Friedler, F.; Fan, L. T.; and Imreh, B. (1998) Process Network Synthesis: Problem Definition, *Networks*, *32*(2) 119-124.

Geoffrion, A.M. (1972) Generalized Benders decomposition. *Journal of Optimization and Theory Application*, *10*(4), 237-260.

Glover, F. (1989). Tabu search-Part I. *ORSA Journal of Computers*, *1*, 190-206.

Glover, F. (1990). Tabu search-Part II. *ORSA Journal of Computers*, *2*, 4-32.

Gmehling, J.; Onken, U.; and Arlt, W. (1977) VLE Data Collection. DECHEMA, Frankfurt.

Gross, B.; and Roosen, P. (1998) Total process optimization in chemical engineering with evolutionary algorithms, *Computers & Chemical Engineering*, *22*, Suppl. S, S229-S236.

Grossmann, I. E. (1985) Mixed-integer programming approach for the synthesis of integrated process flowsheets. *Computers & Chemical Engineering*, *9*(5), 463-482.

Grossmann, I. E. (1996) Mixed-integer optimization techniques for algorithmic process synthesis. *Advances in Chemical Engineering*, *23*, 171-246.

Grossmann, I. E.; and Kravanja, Z. (1995) Mixed-integer nonlinear programming techniques for process systems engineering. *Computers & Chemical Engineering*, *19*, Suppl. S, S189-S204.

Grossmann, I. E.; and Daichendt, M. M. (1996) New trends in optimization-based approaches to process synthesis. *Computers & Chemical Engineering*, *20*(6-7), 665-683.

Grossmann, I. E.; and Türkay, M. (1996) Solution of algebraic systems of disjunctive equations. *Computers & Chemical Engineering*, *20*, S665-S681.

Grossmann, I. E.; and Kravanja, Z. (1997) Mixed-integer nonlinear programming: A survey of algorithms and applications*, The IMA Volumes in Mathematics and its Applications*, Vol.93, Large-Scale Optimization with Applications. Part II: Optimal Design and Control (eds, Biegler, Coleman, Conn, Santosa), Springer Verlag. 73-100.

Grossmann, I. E.; Caballero, J. A.; and Yeomans, H. (1999) Advances in mathematical programming for automated design, integration and operation of chemical processes. *Proceedings of the International Conference on Process Integration PI'99*. Copenhagen, Denmark.

Grossmann, I.E.; and Biegler, L.T. (2004) Part II. Future perspective on optimization. *Computers & Chemical Engineering*, *28*(8), 1193-1218.

Grossmann, I. E.; Aguirre, P. A.; and Barttfeld, M. (2005) Optimal synthesis of complex distillation columns using rigorous models. *Computers & Chemical Engineering*, *29*(6), 1203-1215.

Hallale, N. (1998) Capital cost targets for the optimum synthesis of mass exchange networks, *PhD thesis*, University of Cape Town, Dept. of Chemical Engineering

Hallale, N.; and Fraser, D. M., (2000a) Capital and total cost targets for mass exchange networks, Part I-II., *Computers & Chemical Engineering*, *23*, 1661-1679, 1681-1699.

Hallale, N.; and Fraser, D.M., (2000b) Supertargeting for mass exchange networks, Parts I-II., *Transactions of the Institution of Chemical Engineers*, *78*, Part A, 202-216.

Hendry, I. E.; Rudd, D. F.; and Seader, J. D. (1973) Synthesis in the design of chemical processes. *AIChE Journal*, *19*(1), 1-15.

Hlavacek, V. (1978) Synthesis in design of chemical processes. *Computers & Chemical Engineering*, *2*(1), 67-75.

Hooker, J. N.; Yan, H.; Grossmann, I. E.; and Raman, R. (1994) Logic cuts for processing networks with fixed charges, *Computers & Operations Research*, *21*(3), 265-279.

Hostrup, M.; Gani, R.; Kravanja, Z.; Sorsak, A.; and Grossmann, I. E. (2001) Integration of thermodynamic insights and MINLP optimization for the synthesis, design and analysis of process flowsheets. *Computers & Chemical Engineering*, *25*(1), 73-83.

Hui, C.-W. (1999) Optimising chemical processes with discontinuous function - A novel formulation, *Computers & Chemical Engineering*, *23*, S479-S482.

Jaksland, C. A.; Gani, R.; and Lien, K. M. (1995) Separation processes design and synthesis based on thermodynamic insights. Chemical Engineering Science, 50(3), 511-530.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. (1983) Optimization by simulated annealing. *Science*, *220*(4598), 671-680.

Kister, H.Z. (1992) Distillation design. McGraw-Hill, Inc.

Kocis, G. R.; and Grossmann, I. E. (1987) Relaxation strategy for the structural optimization of process flow sheets, *26*, 1869-1880.

Kravanja, Z.; and Glavic, P. (1997) Cost targeting for HEN through simultaneous optimization approach: A unified pinch technology and mathematical programming design of large HEN. *Computers & Chemical Engineering*, *21*(8), 833-853.

Kravanja, Z.; and Grossmann, I. E. (1997) Multilevel-hierarchical MINLP synthesis of process flowsheets. *Computers & Chemical Engineering*, *21*, Suppl. S, S421-S426.

Lee, S.; and Grossmann, I. E. (2000) New algorithms for nonlinear generalized disjunctive programming. *Computers & Chemical Engineering*, *24*(9-10), 2125-2141.

Lelkes, Z.; Szitkai, Z.; Rév, E.; and Fonyó, Z. (2000) Rigorous MINLP model for ethanol dehydration system. *Computers & Chemical Engineering*, *24*, 1331–1336.

Li, X. N.; and Kraslawski, A. (2004) Conceptual process synthesis: past and current trends. *Chemical Engineering and Processing*, *43*(5), 583-594.

Lin, B.; and Miller, D. C. (2004) Tabu search algorithm for chemical process optimization. *Computers & Chemical Engineering*, *28*(11), 2287-2306.

Linnhoff, B. (1993) Pinch analysis. A state-of-the-art overview. *Transactions of the Institution of Chemical Engineers*, *71*, Part A, 503-522.

Linnhoff, B. (1994) Use pinch analysis to knock down capital costs and emissions. *Chemical Engineering Progress*, *90*(8), 32-57.

Luyben, M. L.; and Floudas, C.A. (1994) Analyzing the interaction of design and control, Part 1: A multiobjective framework and application to binary distillation synthesis. *Computers & Chemical Engineering*, *18*(10), 933-969.

Masso, A. H.; and Rudd, D. F. (1969) The synthesis of systems designs: II. Heuristic structuring. *AIChE Journal*, *15*(1), 10-17.

Mizsey, P.; and Fonyó, Z. (1990) Toward a more realistic overall process synthesis – The combined approach. *Computers & Chemical Engineering*, *14*(11), 1213-1236.

Nishida, N.; Stephanopoulos, G.; and Westerberg, A. W. (1981) A review of process synthesis. *AIChE Journal*, *27*(3), 321-351.

Novak, Z.; Kravanja, Z.; and Grossmann, I. E. (1996) Simultaneous synthesis of distillation sequences in overall process schemes using an improved MINLP approach. *Computers & Chemical Engineering*, *20*(12), 1425–1440.

Pajula, E.; Seuranen, T.; Koiranen, T.; and Hurme, M. (2001) Synthesis of separation processes by using case-based reasoning. *Computers & Chemical Engineering*, 25(4-6), 775-782.

Papalexandri, K. P.; Pistikopoulos, E. N.; and Floudas, C. A. (1994) Mass exchange networks for waste minimization, *Transactions of the Institution of Chemical Engineers*, *72*, Part A, 279-293.

Quesada, I.; and Grossmann, I. E. (1992) An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Computers & Chemical Engineering*, *16*(10-11), 937-947.

Raman, R.; and Grossmann, I. E. (1991) Relation between MILP modelling and logical interference for chemical process synthesis. *Computers & Chemical Engineering*, *15*(2), 73-84.

Raman, R.; and Grossmann, I. E. (1992) Integration of logic and heuristic knowledge in MINLP optimization for process synthesis. *Computers & Chemical Engineering*, *16*, 155-171.

Raman, R.; and Grossmann, I. E. (1993) Symbolic integration of logic in mixed-integer linear programming techniques for process synthesis. *Computers & Chemical Engineering*, *17*, 909-927.

Raman, R.; and Grossmann, I. E. (1994) Modelling and computational techniques for logic based integer programming. *Computers & Chemical Engineering*, *18*, 563-578.

Reneaume, J.M.; Heritier, L.; Domenech, S.; and Joulia, X. (1995) Synthese optimale de reseaux d'echangeurs de chaleur dans l'environement d'un simulateur modulaire. *5éme Congrès Français de Génie des Procédés, Lyon, (19-21 septembre 1995) Collection Récents Progrés en Génie des Procédés, Le Génie des Procédés Complexes*, 9, *42*, 435-440.

Rév, E.; Farkas, T.; and Lelkes, Z. (2005) Process flowsheet structures. *International Journal of Computer Mathematics*, submitted for publication.

Reyes-Labarta, J.A.; and Grossmann, I.E. (2001) Optimal synthesis of liquid–liquid multistage extractors. *Proceedings of ESCAPE-11*. 27–30 May, 2001. Kolding, Denmark, Amsterdam: Elsevier. 487–492.

Rudd, D. F. (1969) The synthesis of system designs: I. Elementary decomposition theory. *AIChE Journal*, *14*(2), 343-349.

Sauar, E.; Ratkje, S. K.; and Lien, K. M. (1996) Equipartition of forces: A new principle for process design and optimization. *Industrial & Engineering Chemistry Research*, *35*(11), 4147-4153.

Seuranen, T.; Hurme, M.; and Pajula, E. (2005) Synthesis of separation processes by case-based reasoning. *Computers & Chemical Engineering*, *29*(6), 1473-1482.

Siirola, J. J. (1996) Strategic process synthesis: Advances in the hierarchical approach. *Computers & Chemical Engineering*, *20*¸ Suppl. B, S1637-S1643.

Smith, R. (1995) Chemical Process Design. McGraw Hill, International Editions.

Szitkai, Z.; Lelkes, Z.; Rév, E.; and Fonyó, Z. (2002) Handling of removable discontinuities in MINLP models for process synthesis problems, formulations of the Kremser equation. *Computers & Chemical Engineering*, *26*(11), 1501–1516.

Szitkai, Z.; Farkas, T.; Lelkes, Z.; Rév, E.; Fonyó, Z.; and Kravanja, Z. (2005) A fairly linear MINLP model for the synthesis of mass exchange networks, *Industrial & Engineering Chemistry Research*, accepted for publication.

The New Encyclopædia Britannica. (1990-1999) 15th edition, Encycl. Britannica Inc., Chicago, Ill.

Türkay, M.; and Grossmann, I. E. (1996) Logic-based MINLP algorithms for the optimal synthesis of process networks, *Computers & Chemical Engineering*, *20*(8), 959-978

Ulmann's Encyclopedia of Industrial Chemistry (1996), Wiley Europe.

Vecchietti, A.; Lee, S.; and Grossmann, I. E. (2003) Modelling of discrete/continuous optimization problems: Characterization and formulation of disjunctions and their relaxations. *Computers & Chemical Engineering*, *27*(3), 433–448.

Viswanathan, J.; and Grossmann, I.E. (1990) A combined penalty-function and outer-approximation method for MINLP optimization. *Computers & Chemical Engineering*, *14*(7), 769-782.

Viswanathan, J.; and Grossmann, I. E. (1993a) An alternate MINLP model for finding the number of trays required for a specified separation objective. *Computers & Chemical Engineering*, *17*(9), 949-955.

Viswanathan, J.; and Grossmann, I. E. (1993b) Optimal feed locations and number of trays for distillation columns with multiple feeds. *Industrial & Engineering Chemistry Research*, *32*(11), 2942-2949.

Watson, I. (1997) Applying case-based reasoning: Techniques for enterprise systems. Morgan Kaufman Publishers, Inc.

Westerlund, T.; and Petterson, F. (1995) An extended cutting plane method for solving convex MINLP problems. *Computers & Chemical Engineering*, *19*, Suppl. S, S131-S136.

Yee, T. F.; and Grossmann, I.E. (1990) Simultaneous optimization models for heat integration II. Heat Exchanger Network Synthesis. *Computers & Chemical Engineering*, *14*(10), 1165-1184.

Yeomans, H.; and Grossmann, I. E. (1999a) A systematic modelling framework of superstructure optimization in process synthesis. *Computers & Chemical Engineering*, *23*, 709-731.

Yeomans, H.; and Grossmann, I. E. (1999b) Nonlinear disjunctive programming models for the synthesis of heat integrated distillation sequences. *Computers & Chemical Engineering*, *23*(9), 1135–1151.

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

Yeomans, H.; and Grossmann, I. E. (2000b) Optimal design of complex distillation columns using rigorous tray-by-tray disjunctive programming models. *Industrial & Engineering Chemistry Research*, *39*(11), 4326-4335.

Abbreviations and notations

Abbreviations

BDIMR	Binarily Decreased and Ideal MINLP Representation
BGR	Basic GDP Representation
BMIMR	Binarily Minimal and Ideal MINLP Representation
BMR	Basic MINLP Representation
BMMR	Binarily Minimal MINLP Representation
CBR	Case-Based Reasoning
CMR	Conventional MINLP Representation
CNF	Conjunctive Normal Form
DNF	Disjunctive Normal Form
GDP	Generalized Disjunctive Programming
HEN	Heat Exchange Network
HENS	Heat Exchange Network Synthesis
IMR	Ideal MINLP Representation
ILP	Integer Linear Programming
LP	Linear Programming
MEN	Mass Exchange Network
MENS	Mass Exchange Network Synthesis
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Nonlinear Programming
MR	MINLP Representation
MSA	Mass Separating Agent
MSG	Maximal Structure Generation
NLP	Nonlinear Programming
NSXMR	Non-considered Structures eXcluded MINLP Representation
OTOE	One Task-One Equipment
SEN	State-Equipment Network
SSG	Generation of Solution-Structure
STN	State-Task Network
TAC	Total Annual Cost
VTE	Variable Task-Equipment

Notations

Variables and parameters

a	attribute in general	
A	cross section of the column	[m^2]
A	Antoine constant A (parameter)	[-]
B	Antoine constant B (parameter)	[-]
Bot	bottom product component flowrate	[kmol/hr]
c	cost	[USD, USD/yr, or USD/kJ]
C	total cost	[USD/yr]
C	Antoine constant C (parameter)	[-]
d	design and control variable	
DC	column diameter	[m]
dF	feed stream component flowrate	[kmol/hr]
Dis	distillate component flowrate	[kmol/hr]
dL	liquid stream component flowrate between units	[kmol/hr]
dV	vapor stream component flowrate between units	[kmol/hr]
e	extensive variable	
f	fugacity	[Hgmm]
F	feed component flowrate	[kmol/hr]
$F_{\max}$	F-factor	[$\sqrt{Pa}$]
$Feed$	feed flow rate	[kmol/hr]
hB	molar enthalpy of the bottom product	[kJ/kmol]
hD	molar enthalpy of the distillate	[kJ/kmol]
hdF	component molar enthalpy of feed stream	[kJ/kmol]
hdL	component molar liquid enthalpy of stream between units	[kJ/kmol]
hdV	component molar vapor enthalpy of stream between units	[kJ/kmol]
hF	molar enthalpy of the feed	[kJ/kmol]
ΔH	latent heat	[kJ/kmol]
hL	molar liquid enthalpy for non-transport units	[kJ/kmol]
htL	molar liquid enthalpy for transport units	[kJ/kmol]
htV	molar vapor enthalpy for transport units	[kJ/kmol]
hV	molar vapor enthalpy for non-transport units	[kJ/kmol]
i	intensive variable	

l	index of graphs	[-]
L	lower bound in Chapter 1 and Chapter 4	
L	liquid component flowrate for non-transport units in Chapter 5	[kmol/hr]
LIQ	total liquid flowrate	[kmol/hr]
m	normalized molar mass	[-]
$\dot{m}$	amount flowrate	[kg/s]
M	molar mass in Chapter 3	[g/mol]
M	Big M parameter	
n	number of items denoted in the subscript (parameter)	[-]
N	number of items denoted in the subscript (variable)	[-]
NI	non-ideality	[-]
o	operation variable	
OBJ	objective value	
P	pressure	[Hgmm]
q	enthalpy state	[-]
Q	integer variable in Binarily Minimal Representation	[-]
QC	effective heat transfer in condenser	[kJ/hr]
QR	effective heat transfer in reboiler	[kJ/hr]
Ref	reflux ratio	[-]
sc	similarity value of components	[-]
sim	local similarity	[-]
SIM	global similarity	[-]
t	normalized temperature	[-]
T	temperature	[K]
tL	liquid component flowrate for transport units	[kmol/hr]
tV	vapor component flowrate for transport units	[kmol/hr]
U	upper bound	
UF	update factor	[-]
$UNITS$	number of membrane modules in a section	[-]
V	vapor component flowrate for non-transport units	[kmol/hr]
VAP	total vapor flowrate	[kmol/hr]
w	weight	
x	binary variable in Chapter 1 and Chapter 4	[mol/mol]

x	liquid mole fraction in Chapter 1 and Chapter 5	[mol/mol]
y	vapor mole fraction	[mol/mol]
z	binary variable	
$\tilde{z}$	binary variable in Binarily Minimal Representation	[-]
Z	logical variable	
α	number of input ports	[-]
β	number of output ports	[-]
β_{tax}	tax factor	[-]
γ	activity coefficient	[-]
Δ	difference variable	
ε	small positive value	[-]
φ	fraction of stream	[-]
ξ	recovery	[mol/mol]
ρ	density	[kg/m^3]
λ	purity requirement, upper bound	[mol/mol]
τ	purity requirement, lower bound	[mol/mol]

Subscripts

a	attribute
b	boiling point
bv	binary variables
c	component
cg	considered graphs
ch	charge
cond	conditional
e	edge
f	feed
g	graphs
im	units not present
ip	units present
j	equilibrium stage

k	unit containing equilibrium stages
l	graphs
m	molar mass in Chapter 3
m	unit in Chapter 1 and Chapter 4
ma	membrane section
max	greatest value in the data base in Chapter 3
max	maximum number of modules and sections in Chapter 4
mb	membrane module in a section
min	smallest value in the data base in Chapter 3
p	product
perm	permanent
r	port
rg	represented graphs
s	column section
st	equilibrium stage
t	boiling point in Chapter 3
t	type of unit in Chapter 4
V	vapor

Superscripts

0	vapor pressure
B	bottom product
bu	boil-up
C	condenser
cond	condensation
CW	cooling water
D	distillate
F	feed
first	first stage
fix	fix
in	inlet
inner	inner stages
L	liquid

last	last stage
LPS	low pressure steam
max	maximal value
out	outlet
R	reboiler
ref	reflux
S	source case
T	target case in Chapter 3
T	transport unit in Chapter 5
U	conditional unit containing equilibrium stages
UP	upper bound
V	vapor
vap	vaporization
var	variable

Vectors

d	distance vector
e	basis vector
S	attribute vector of source case
T	attribute vector of target case
x	composite vector
ε	non-zero vector of small length

in other cases the vector of a variable has the same symbol as the variable in bold style

Sets and regions

B	subset of the feasible region
C	set of components
E	set of edges
FR	feasible region
IO	set of indices for which $\tilde{z}_i$=0
I1	set of indices for which $\tilde{z}_i$=1

J_n	sets of equilibrium stages in a unit; n is equal to the number of equilibrium stages in the particular unit
K	set of conditional units in a column section
M	set of units
R	set of graphs in Chapter 4
R	set of real numbers in Chapter 1 and Chapter 4
S	set of column sections /1=lower, 2=upper/
X	region of continuous variables
Z	region of binary variables
$\overline{Z}$	region of logical variables

Functions

f	function in general
g	function in general
h	function in general
P	function of unit operations
Pfix	function of fix cost
Pvar	function of variable cost
Φ	function in general
Ω	logical truth function

Appendix

MINLP representation of Kocis and Grossmann (1987)

$$\min C = z_I + 1.5z_{II} + 3.5z_{III} + 1.8a_1 + 7.0b_1 + b_2 + 1.2b_3 - 11.0c$$

$$s.t. \quad b_2 - \ln(1+a_2) = 0$$

$$b_3 - 1.2\ln(1+a_3) = 0$$

$$c - 0.9b = 0$$

$$b_1 + b_2 + b_3 - b = 0$$

$$a_1 - a_2 - a_3 = 0$$

$$a_2 - 5z_I \le 0$$

$$a_3 - 5z_{II} \le 0$$

$$b - 5z_{III} \le 0$$

$$c \le 1$$

$$b_2 \le 5$$

$$z_I, z_{II}, z_{III} \in \{0,1\}$$

$$a_1, a_2, a_3, b, b_1, b_2, b_3, c \ge 0$$

Thermodynamic constants

Table A1. Antoine constants in Example 3.3

	A_c	B_c	C_c
heptane	6.89386	1264.370	216.640
toluene	6.95087	1342.31	219.187

Table A2. Antoine constants in Example 5.1

	A_c	B_c	C_c
benzene	6.87987	1196.76	219.161
toluene	6.95087	1342.31	219.187

Table A3. Antoine constants in Example 5.2

	A_c	B_c	C_c
methanol	8.08097	1582.271	239.726
propanol	8.37895	1788.02	227.438
buthanol	7.838	1558.19	196.881

Table A4. Antoine constants in Example 5.3

	A_c	B_c	C_c
ethanol	8.11220	1592.864	226.184
water	8.07131	1730.630	233.426

Table A5. Margules parameters in Example 5.3

A12 (ethanol-water)	1.5871
A21 (water-ethanol)	0.7941

Declaration

I declare that no portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Nyilatkozat

Alulírott *Farkas Tivadar* kijelentem, hogy ezt a doktori értekezést magam készítettem és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2005. december 20.

Farkas Tivadar