Discrete Optimization Methods and Their Role in the Integration of Planning and Scheduling

Article *in* Chemical Engineering -New York- Mcgraw Hill Incorporated then Chemical Week Publishing Llc- · April 2001

3 authors, including:

Iiro Harjunkoski
ABB AG, Corporate Research Germany

Discrete Optimization Methods and their Role in the Integration of Planning and Scheduling

Ignacio E. Grossmann*, Susara A. van den Heever and Iiro Harjunkoski
Department of Chemical Engineering
Carnegie Mellon University, Pittsburgh, PA 15213

March 1, 2001

Abstract
The need for improvement in process operations, logistics and supply chain management has created a great demand for the development of optimization models for planning and scheduling. In this paper we first review the major classes of planning and scheduling models that arise in process operations, and establish the underlying mathematical structure of these problems. As will be shown, the nature of these models is greatly affected by the time representation (discrete or continuous), and is often dominated by discrete decisions. We then briefly review the major recent developments in mixed-integer linear and nonlinear programming, disjunctive programming and constraint programming, as well as general decomposition techniques for solving these problems. We present a general formulation for integrating planning and scheduling to illustrate the models and methods discussed in this paper.

Key words: planning, scheduling, optimization, mixed-integer programming.

1. Introduction

The development of optimization models for planning and scheduling of chemical processes has received significant attention over the last 5-7 years. One major reason has been the realization by industry that large potential savings can be achieved by improving the logistics of manufacturing in chemical processes. Examples of savings include lower inventories, lower transition costs, and reduction in production shortfalls. The interest in planning and scheduling has further increased with industry's goal of improving the management and dynamics of their supply chains. Finally, major advances in large-scale computation and mathematical programming have promoted the interest in applying these techniques to planning and scheduling problems.

The goal of this paper is to provide an overview of the optimization based models for planning and scheduling, review the solution strategies and mathematical programming methods that are available for solving these problems, and propose a conceptual model for integrating planning and scheduling. Finally, we present three examples to illustrate the application of some of the techniques discussed in this paper.

2. Review on Planning and Scheduling

Both planning and scheduling deal with the allocation of available resources over time to perform a collection of tasks. In the context of process systems, planning and scheduling refer to the strategies of allocating equipment and utility or manpower resources over time to execute processing tasks required to manufacture one or several products. The difference between planning and scheduling is not always clear cut. However, in general the difference is that planning deals with longer time horizons (e.g. weeks, few months) and is largely concerned with high level decisions such as investment in new facilities and production levels. Scheduling on the other hand deals with shorter time horizons (e.g. days, few weeks) with the emphasis often being on the lower level decisions such as sequencing of operations. Also, in planning maximization of profit usually plays a major role, while in scheduling the emphasis tends to be on feasibility for fulfilling a given number of orders, or on completing the required tasks in the shortest time.

* Author to whom correspondence should be addressed. e-mail: grossmann@cmu.edu.

Hence, economics tends to play a greater role in planning than in scheduling. It should be noted, however, that the distinction between planning and scheduling is becoming increasingly blurred by the capability of optimizing simultaneous planning and scheduling decisions, particularly in the context of supply chain optimization problems.

Planning

A detailed review of planning and scheduling is out of the scope of this paper. In this section we therefore focus on pointing the reader to some useful papers covering both specific planning problems and reviews on some problem classes, as well as a general discussion on the nature of planning problems. While no one review covers all types of planning problems, some reviews can be found in the operations research literature for specific types of planning problems. Erengüç *et al.* (1999) review work on the integrated production and distribution planning of supply chains. They discuss the different stages of the supply chain, give some general formulations and critically evaluate the relevant literature from the operations research community. Other reviews discuss planning models for freight transportation (Crainic and Laporte, 1997), optimization methods for electric utility resource planning (Hobbs, 1995), and strategic facility location methods that consider either stochastic or dynamic problem characteristic (Owen and Daskin, 1998). In the chemical engineering literature, a review on literature for single- and multisite planning and scheduling can be found in Shah (1998), while reviews on planning and scheduling literature for batch/semicontinuous plants can be found in Reklaitis (1991, 1992) and Rippin (1993).

Planning problems can mainly be categorized as strategic, tactical or operational, depending on the decisions involved and the time horizon under consideration. Strategic planning covers the longest time horizons in the range of one to several years and decisions cover the whole width of the organization, while focussing on major investments. Examples of strategic planning problems include facility location problems (e.g. Mazzola and Neebe, 1999), hydrocarbon well platform investment planning (e.g. Iyer *et al*., 1998; Van den Heever and Grossmann, 2000), and longterm planning of process networks (e.g. Sahinidis *et al*., 1989) where it is essential to consider the far future in making big investment decisions. Tactical planning typically covers the midterm horizon of between a few months to a year and decisions cover issues such as production, inventory and distribution. Midterm production planning or supply chain planning is a good example of tactical planning (e.g. Bok *et al*., 2000; McDonald and Karimi, 1997; Perea *et al*., 2000; Dimitriadis *et al*., 1997). Operational planning usually covers a horizon of one week to three months and involves decisions regarding the actual operations and resource allocation. Applications include the operational planning of utility systems (e.g. Iyer and Grossmann, 1998a) and the planning of refinery operations (e.g. Moro *et al*., 1998). On this level, planning decisions are often closely related to scheduling decisions and it becomes more important to integrate these. In the past planning and scheduling issues have mostly been addressed separately or sequentially for reasons of complexity, and only recently have simultaneous planning and scheduling approaches emerged. Birewar and Grossmann (1990) proposed a model for the simultaneous planning and scheduling of multipurpose batch plants, while Shah and Pantelides (1991) presented a model for simultaneous campaign formation and planning. Papageorgiou and Pantelides (1996a, 1996b) address the issue in a two part article proposing a mathematical formulation and decomposition approach for integrated campaign planning and scheduling of multipurpose batch/semicontinuous plants. We address the integration of planning and scheduling in Section 4 through a generalized disjunctive model.

In terms of uncertainties, planning models have either a deterministic or stochastic nature. Deterministic models assume predictions for prices, demands and availabilities to be known with certainty. These models are often sufficient for short-term planning and scheduling, but when longer time horizons are considered incorporating uncertainty directly becomes more important. However, deterministic models are still useful even when uncertainty needs to be incorporated, since they can be used to analyze different scenarios for the uncertain parameters without the additional complexity associated with stochastic models. In addition, deterministic models form the basis for stochastic models that include uncertainty through scenarios. Stochastic models include uncertainty either by explicit probability distributions or by scenarios, and often require specialized solution methods due to their complexity. A vast number of articles have been published in the area of process planning under uncertainty and a complete list of all relevant ones cannot be provided within the scope of this paper. We therefore refer readers to some recent publications: Liu and

Sahinidis (1996) proposed a two-stage stochastic programming approach for process planning under uncertainty. These authors first consider discrete random parameters and develop a Benders-based decomposition algorithm for the solution. They then continue to show the applicability of their approach to continuous random variables. Ierapetritou *et al*. (1996) discuss modeling issues in design and planning under uncertainty and propose a decomposition algorithm for a combined multiperiod/stochastic programming problem. Clay and Grossmann (1997) consider planning problems with uncertainty in both demands and cost coefficients, and represent these uncertainties by finite discrete probability distribution functions. They also propose an iterative aggregation/disaggregation algorithm that is highly parallel for the solution of this two-stage stochastic programming problem. Ahmed and Sahinidis (1998) propose a linear method of incorporating robustness of the second stage decisions into two-stage models for process planning under uncertainty and present a heuristic solution approach, but emphasize the need to exact solution methods for this model. Rather than using a stochastic optimization framework, Applequist *et al.* (2000) proposed a risk measure for supply chain optimization. Some significant progress has also been made in incorporating uncertainty into combined planning and scheduling models. Petkov and Maranas (1997) extend the combined planning and scheduling model first proposed by Birewar and Grossmann (1997) to include demand uncertainties, and express the stochastic elements in equivalent deterministic forms to yield solution times comparable to pure deterministic models. A scenario-based approach to incorporate uncertainty at the planning level for an online scheduler of a multiproduct batch plant was proposed by Sand *et al.* (2000). Their model has two hierarchical levels, where uncertainty at the planning level is incorporated explicitly in the upper level. While the above mentioned approaches show significant progress in incorporating uncertainty into large-scale planning and scheduling models, the characterization of uncertainties and development of efficient solution algorithms remain complex and challenging problems.

Scheduling

Comprehensive reviews on scheduling can be found in Rippin (1993) who addresses the general status of batch processing systems engineering with emphasis in design, planning and scheduling. Reklaitis (1991, 1992) presents a comprehensive review of scheduling and planning of batch process operations. His main focus is to describe the basic components of the scheduling problem and review the existing solution methods. Pekny and Zentner (1994) summarize the basic scheduling technology with association to the advances in computer technology. Grossmann *et al.* (1996) provide an overview of mixed integer optimization techniques for the design and scheduling of batch processes, with emphasis on general purpose methods for mixed integer linear (MILP) and mixed integer nonlinear (MINLP) problems. Pinto and Grossmann (1998) present a classification of scheduling problems, and characterize the major types of integer and mixed-integer constraints that arise for the assignment and sequencing decisions. Shah (1998) presents an overview of single and multisite scheduling methods, while Pekny and Reklaitis (1998) provide a review in terms of the computational complexity that is involved in scheduling problems.

A major difficulty that has been faced in the area of scheduling is that there is a great diversity of problems that have tended to prevent the development of unified solutions. To appreciate this issue, consider Figure 1 from Pinto and Grossmann (1998) that presents a road map for classifying scheduling problems. Equipment similarity and unit connectivity define the topology of the plant. In serial plants, products follow the same production path, therefore it is possible to recognize a specific direction in the plant floor. Networks of arbitrary topology tend to occur when products have low recipe similarity and/or when equipment is interconnected. Most methods do not handle mass balances explicitly; instead, production is represented by batches (or lots). Products follow a series of tasks, which are collections of elementary chemical and physical processing operations. Note the close relationship between the plant topology and the sequence of tasks for products: if all products follow the same sequence of tasks it is usually possible to define processing stages in the plant, defined as processing equipment that can perform the same operations. Moreover, lot sizes can be variables, such as in the case of the lot-sizing problem, or fixed parameters.

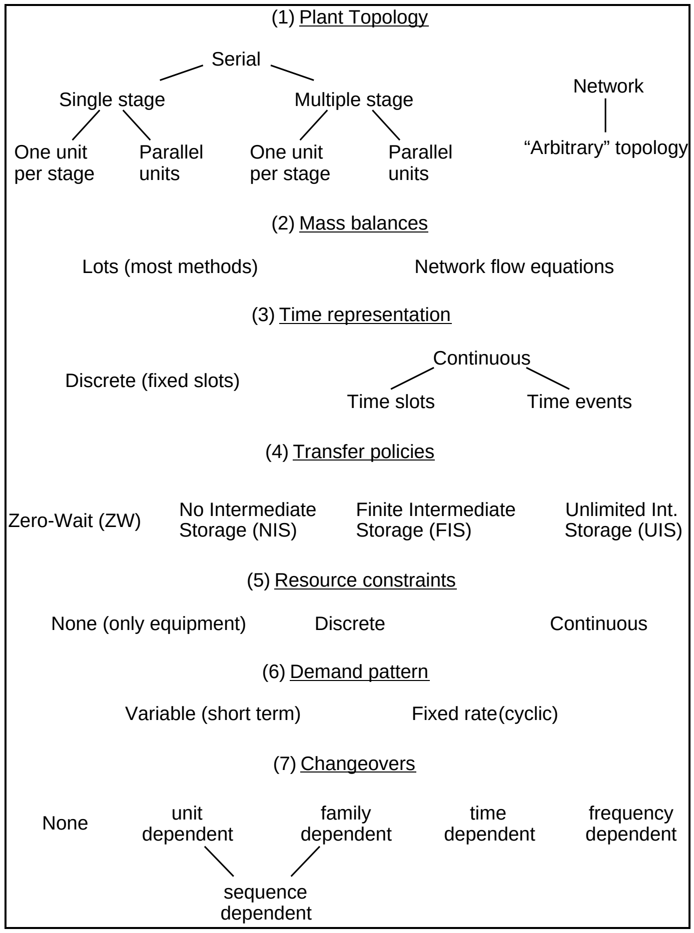

Figure 1. Classification and road map scheduling problems.

A major issue in modeling scheduling problems concerns the time domain representation. The most general is a continuous time domain representation that makes use of either time slots of variable length, or time events. If a discrete time representation is adopted, slots have equal and fixed duration. In this case there is the need to use a sufficiently large number of slots in order to have a suitable approximation of the original problem. An advantage, however, with discrete time domain is that it is much easier to handle resource constraints or track inventory levels. In continuous time formulations it is usually possible to postulate a much smaller number of time slots or time events reducing the problem size, although often at the expense of introducing nonlinearities in the model.

Another major issue in plant scheduling deals with the presence of intermediate storage. There are four different transfer policies: Zero-Wait (ZW), No-Intermediate-Storage (NIS), Finite-Intermediate-Storage (FIS) and Unlimited-Intermediate-Storage (UIS) (Ku *et al.*, 1987). It is important to note that FIS corresponds to the most general case. Nevertheless, the main advantage of the remaining three cases is that there is no need to model inventory levels. In the scheduling of a process plant, processing tasks require utilities and manpower. Utilities may include, for example, steam, electricity, cooling water, etc. In some scheduling applications, apart from equipment, finite resources that are limited are required for these process tasks. Resource constrained scheduling problems are inherently difficult, due to the fact that besides the efficient allocation of units to meet product demands, it is also necessary to consider the feasible grouping of simultaneously executed tasks so as to utilize resources within their availability limits.

Short term scheduling is relevant to plants that must satisfy individual customer orders with varying demand patterns. In this case, product requirements are given as a set of orders, where each order has associated with it a certain product, the amount and a due date. In contrast, cyclic scheduling is relevant for plants operating with a stable market in which the product demands are given as constant rates. This allows a more simplified plant operation in which the same production sequence is executed repeatedly with a fixed frequency. When switching between products, or even after one or more batches of the same product, units may require cleaning and setup for safety and/or product quality. Changeover requirements depend on the nature of the units and the products in the plant. Sequence dependent changeovers represent the most general and difficult situation, in which every pair of consecutive operations may give rise to different time and/or cost requirements. The need for unit setup may be expressed in terms of the frequency of utilization. For instance, a changeover may be needed after every batch or after a certain number of batches, regardless of the nature of the products. In the case of time dependent cleaning, there is a maximum time interval during which a unit may be utilized.

From all the scheduling models that have been proposed in the chemical engineering literature, the most general model is the one by Kondili *et al*. (1993), which addresses short term scheduling of batch operations. Major capabilities of this multiperiod MILP model include the following: (a) assignments of equipment to processing tasks need not be fixed, (b) variable size batches can be handled with the possibility of mixing and splitting, (c) different intermediate storage and transfer policies can be accommodated, as well as limitations of resources. In the work by Kondili *et al* (1993) a major assumption that was made is that the time domain can be discretized in intervals of equal size, which in practice often means having to perform some rounding to the original data. In addition, changeover times are usually neglected since they cannot be easily handled by this model. The key aspect in the MILP model by Kondili *et al.* (1993), is the state-task network (STN) representation. This network has two types of nodes: (a) state nodes that correspond to feeds, intermediates and final products; (b) task nodes that represent processing steps. Figure 2 presents an example of a state task network. It should be noted that the equipment is considered separately. In general it is assumed that each unit can perform several of the tasks in the STN network. The resulting MILP model determines the timing of the operations, assignments of equipment to operations, and flow of material through the network. The objective is to maximize a given profit function. Figure 3 shows the results of the optimal schedule of the example in Figure 2. It should be noted that the reformulation by Shah *et al*. (1993) led to a significant improvement in the LP relaxation of the MILP, with which fairly large problems can be solved. Furthermore, Pantelides (1994) proposed the Resource Task Network (RTN) representation, which leads to a more compact model than the STN, although it is actually equivalent. It is interesting to note that in the context of Figure 1, both the STN and RTN models can handle networks with arbitrary topology, handle flow equations for the mass balances, are based on discrete time representation, can handle all types of transfers and resource constraints, and deal with short term variable demands. Continuous time versions of this model have been proposed for instance by Zhang and Sargent (1996), Mockus and Reklaitis (1996), and Ierapetritou and Floudas (1998).

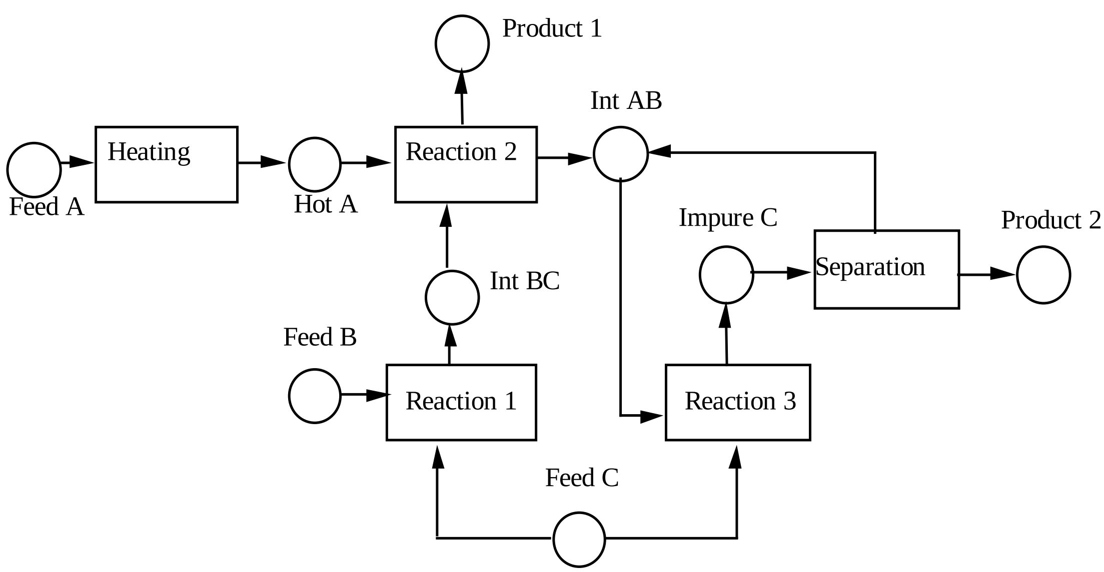

Figure 2. State-task network for numerical example**.**

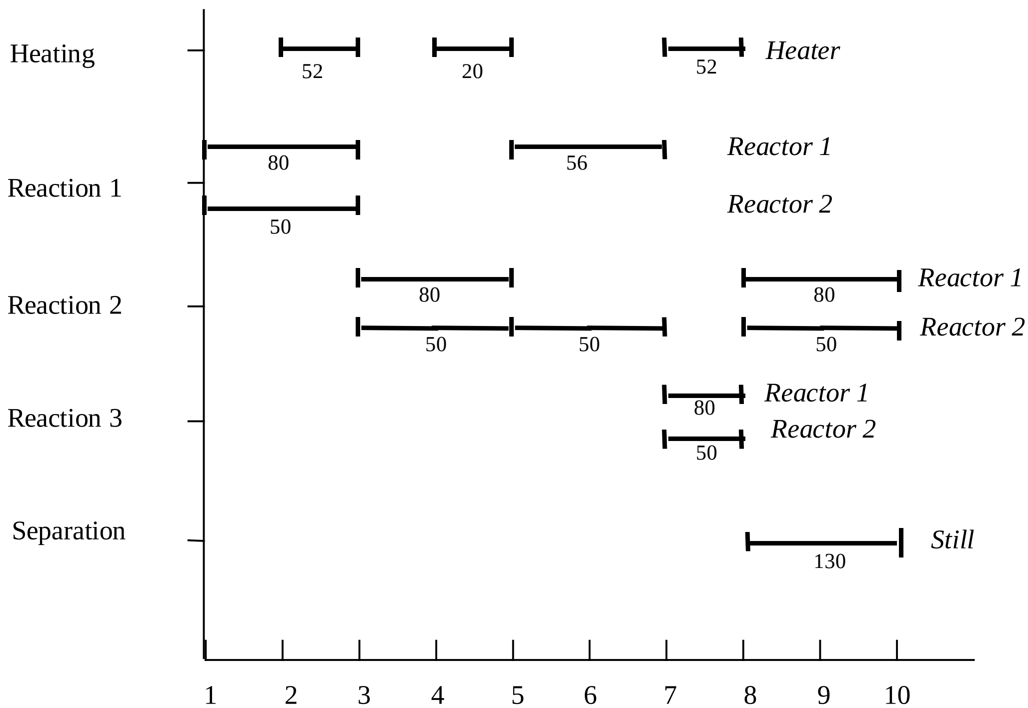

Figure 3. Optimal schedule for network in Figure 2.

3. Mathematical Programming

Planning and scheduling problems generally give rise to discrete/continuous optimization problems and we therefore find a discussion on the major mathematical programming techniques appropriate in the current context. When these optimization problems are represented in algebraic form, they correspond to mixed-integer optimization problems that have the following form:

$$\begin{aligned} &\min Z = f(x,y) \\ &s.t. \quad h(x,y) = 0 \\ &\qquad\; g(x,y) \le 0 \\ &\qquad\; x \in X, y \in \{0,1\}^m \end{aligned} \qquad \text{(MIP)}$$

where $f(x, y)$ is the objective function (e.g. cost), $h(x, y) = 0$ are the equations that describe the performance of the system (material balances, production rates), and $g(x, y) \le 0$ are inequalities that define the specifications or constraints for feasible plans and schedules. The variables x are continuous and generally correspond to state variables, while y are the discrete variables, which generally are restricted to take 0-1 values to define for instance the assignments of equipment and sequencing of tasks. Problem (MIP) corresponds to a mixed-integer nonlinear program (MINLP) when any of the functions involved are nonlinear. If all functions are linear it corresponds to a mixed-integer linear program (MILP). If there are no 0-1 variables, the problem (MIP) reduces to a nonlinear program (NLP) or linear program (LP) depending on whether or not the functions are linear.

The formulation and solution of major types of mathematical programming problems can be effectively performed with modeling systems such as GAMS (Brooke at al., 1992), and AMPL (Fourer *et al*., 1992). While these require that the model be expressed explicitly in algebraic form, they have the advantage that they automatically interface with codes for solving the various types of problems. They also perform automatic differentiation and allow the use of indexed equations, with which large scale models can be readily generated. It should also be noted that these modeling systems are now widely available on desktop PCs. We review the major classes of mathematical programming models in the following paragraphs.

Linear and Mixed-Integer Programming

These are without a doubt the types of models that are most frequently encountered for planning and scheduling. The reason is that these models involve in most cases discrete time representations coupled with fairly simple performance models. While in the past most models were LPs, most of them are nowadays MILPs due to the discrete decisions that are involved in investment, expansion and operation for planning, and assignment and sequencing decisions for scheduling.

Mixed-integer linear programming problems have the general form:

$$\begin{aligned} &\min Z = a^T x + b^T y \\ &s.t. \quad Ax + By \leq d \\ &\qquad\; x \geq 0, y \in \{0,1\}^m \end{aligned} \qquad \text{(MILP)}$$

For the case when no discrete variables *y* are involved, the problem reduces to a linear programming (LP) problem. This is a special class of convex optimization problems for which the optimal solution lies at a vertex of the polytope defined by the inequalities $Ax \leq d$. The solution of LP problems relies largely on the simplex algorithm (Chvatal, 1983; Saigal, 1995), although lately interior-point methods (Marsten *et al*, 1990) have received increased attention for solving very large problems because of their polynomial complexity. MILP methods rely largely on simplex LP-based branch and bound methods (Nemhauser and Wolsey, 1988) that consists of a tree enumeration in which LP subproblems are solved at each node, and eliminated based on bounding properties. These methods are being improved through cutting plane techniques (Balas *et al*., 1993), which produce tighter lower bounds for the optimum. LP and MILP codes are widely available. The best known include CPLEX (ILOG, 2000), OSL (IBM, 1992) and XPRESS (Dash Associates, 1999), all which have achieved impressive improvements in their problem solving capabilities. It is worth noting that since MILP problems are NP-complete it is always possible to run into time limitations when solving problems with a large number of 0-1 variables, especially if the integrality gap (difference between optimal integer objective and optimal LP relaxation) is large.

Nonlinear Programming

NLP models have the advantage over LP models of being able to explicitly handle nonlinearities and are largely used for real-time optimization. These models only involve continuous variables and are fairly restrictive for planning and scheduling, although they are important subproblems in MINLPs. NLP problems correspond to continuous optimization problems that can be expressed as follows:

$$\begin{aligned} &\min Z = f(x) \\ &s.t. \quad h(x) = 0 \\ &\qquad\; g(x) \leq 0 \\ &\qquad\; x \in X \end{aligned} \qquad \text{(NLP)}$$

Provided the functions are continuous and differentiable, and certain constraint qualifications are met, a local optimum solution to problem (NLP) is given by the Karush-Kuhn-Tucker conditions (Minoux, 1983). The solution of NLP problems (Fletcher, 1987; Bazaara *et al*. 1994), relies either on the successive quadratic programming algorithm (SQP) (Han, 1976; Powell, 1978; Schittkowski, 1981), or on the reduced gradient method (Murtagh and Saunders, 1978, 1982). Major codes include MINOS and CONOPT (Drud, 1994) for the reduced gradient method, and OPT (Vasantharajan *et al.*, 1990) for the SQP algorithm. These NLP methods are guaranteed to find the global optimum if the problem is convex (i.e. convex objective function and constraints). When the NLP is nonconvex a global optimum cannot be guaranteed. One option is to try to convexify the problem, usually through exponential transformations, although the number of cases where this is possible is rather small. Alternatively, one could use rigorous global optimization methods, which over the last few years have made significant advances. These methods assume that special structures are present in the problem, such as bilinear, linear fractional and concave separable functions. Although this may appear to be quite restrictive, Smith and Pantelides (1996) have shown that algebraic models are always reducible to these structures, provided they do not involve trigonometric functions. For a general review on global optimization see Horst and Tuy (1993), Horst and Pardalos (1995) and Floudas (2000). Recent developments in chemical engineering can be found in Grossmann (1996). Computer codes for global optimization still remain in the academic domain, and the best known are BARON by Sahinidis and Ryoo (1995), and α-BB by Floudas *et al.* (1996). It should also be noted that non-rigorous techniques which have also become popular, such as simulated annealing (Kirkpatrick *et al.*, 1983) and genetic algorithms (Goldberg, 1989), do not make any assumptions on the functions, but cannot guarantee rigorous solutions in a finite amount of time. Furthermore, these methods do not formulate the problem as a mathematical program since they involve procedural search techniques that in turn require some type of discretization, and the violation of constraints is handled through ad-hoc penalty functions.

Mixed-integer nonlinear programming

MINLP models typically arise in planning and also in scheduling when using continuous time representations, particularly for cyclic policies and for nonlinear performance models. The most common form of MINLP problems is a special case of problem (MIP), in which the 0-1 variables are linear while the continuous variables are nonlinear:

$$\begin{aligned} &\min Z = c^T y + f(x) \\ &s.t. \quad h(x) = 0 \\ &\qquad By + g(x) \le 0 \\ &\qquad x \in X,\, y \in \{0,1\}^m \end{aligned} \tag{MINLP}$$

Major methods for MINLP problems include first Branch and Bound (BB) (Gupta and Ravindran, 1985; Borchers and Mitchell, 1994; Stubbs and Mehrotra, 1996), which is a direct extension of the linear case, except that NLP subproblems are solved at each node. Generalized Benders Decomposition (GBD) (Benders, 1962; Geoffrion, 1972), and Outer-Approximation (OA) (Duran and Grossmann, 1986; Yuan, Zhang, Piboleau and Domenech, 1988; Fletcher and Leyffer, 1994; Ding-Mai and Sargent, 1992), are iterative methods that solve a sequence of alternate NLP subproblems with all the 0-1 variables fixed, and MILP master problems that predict lower bounds and new values for the 0-1 variables. The difference between the GBD and OA methods lies in the definition of the MILP master problem; the OA method uses accumulated linearizations of the functions, while GBD uses accumulated Lagrangean functions parametric in the 0-1 variables. The LP/NLP based branch and bound by Quesada and Grossmann (1992) essentially integrates both subproblems within one tree search, while the Extended Cutting Plane Method (ECP) (Westerlund and Pettersson, 1995) does not solve the NLP subproblems, and relies exclusively on successive linearizations. All these methods assume convexity to guarantee convergence to the global optimum. Nonrigorous methods for handling nonconvexities include the equality relaxation algorithm by Kocis and Grossmann (1987) and the augmented penalty version of it (Viswanathan and Grossmann, 1990). A review on these methods and how they relate to each other can be found in Grossmann and Kravanja (1997). The only commercial code for MINLP is DICOPT (OA-GAMS), although there are a number of academic versions (MINOPT by Floudas and co-workers, α-ECP by Westerlund and co-workers). Tawarmalani and Sahinidis (2000) have recently expanded their BARON global optimization code to MINLP problems through a number of extensions of the above methods.

Logic-based optimization

In recent years a new trend that has emerged is to formulate and solve discrete/continuous optimization problems with logic-based optimization models and methods. These methods, which facilitate problem formulation and often reduce the combinatorial search, are starting to have a significant impact in planning and scheduling problems. The two major methods are Generalized Disjunctive Programming (GDP) (Raman and Grossmann, 1994) and Constraint Programming (Van Hentenryck, 1989).

Generalized Disjunctive Programming. The basic idea in GDP models is to use Boolean and continuous variables, and formulate the problem with an objective function, subject to three types of constraints: (a) global inequalities that are independent of discrete decisions; (b) disjunctions that are conditional constraints involving an OR operator; (c) pure logic constraints that involve only the Boolean variables. More specifically, the problem is given as follows:

$$
\begin{aligned}
&\min Z = \sum_{k \in K} c_k + f(x) \\
&s.t. \\
&\qquad g(x) \le 0 \\
&\qquad \bigvee_{j \in I_k} \begin{bmatrix} y_{jk} \\ h_{jk}(x) \le 0 \\ c_k = \gamma_{jk} \end{bmatrix} \quad k \in K \\
&\qquad \Omega(y) = True \\
&\qquad x \in X,\ y_{jk} \in \{True, False\}
\end{aligned}
\tag{GDP}
$$

where x are continuous variables and y are the Boolean variables. The objective function involves the term $f(x)$ for the continuous variables (e.g. operating cost) and the charges c_k that depend on the discrete choices. The equalities/inequalities $g(x) \le 0$ must hold regardless of the discrete conditions, and $h_{jk}(x) \le 0$ are conditional constraints that must be satisfied when the corresponding Boolean variable y_{jk} is True for the j^{th} term of the k^{th} disjunction. The set I_k represents the number of choices for each disjunction defined in the set K. Also, the fixed charge c_k is assigned the value γ_{jk} for that same variable. Finally, the constraints $\Omega(y)$ involve logic propositions in terms of Boolean variables.

Problem (GDP) represents an extension of disjunctive programming (Balas, 1985), which in the past has been used as a framework for deriving cutting planes for the algebraic problem (MIP). It is interesting to note that any GDP problem can be reformulated as an MIP problem, and vice-versa. It is more natural, however, to start with a GDP model, and reformulate it as an MIP problem. This is accomplished by reformulating the disjunctions using the convex hull transformation (Türkay and Grossmann, 1996) or with "big-M" constraints. The propositional logic statements are reformulated as linear inequalities (Raman and Grossmann, 1991; 1994). For the linear case of problem GDP, and when no logic constraints are involved, Beaumont (1991) proposed a branch and bound method that does not rely on 0-1 variables and branches directly on the equations of the disjunctions. This method was shown to outperform the solution of the alternative algebraic MILP models. Raman and Grossmann (1994) developed a branch and bound method for solving problem GDP in hybrid form, i.e. with disjunctions and mixed-integer constraints. For this they introduced the concept of "w-MIP representability" to denote those disjunctive constraints that can be transformed into mixed-integer form without loss in the quality of the relaxation. Hooker and Osorio (1996) developed a different branch and bound method which in a way is a generalization of Beaumont's method in that it does not introduce 0-1 variables, and addresses problems directly in the form of the GDP problem.

For the nonlinear case of problem (GDP), Lee and Grossmann (1999) have developed reformulations and algorithms that rely on obtaining the algebraic description of the convex hull of the nonlinear convex inequalities. The reformulations lead to tight MINLP problems, while the algorithms generally involve branch and bound methods where branching is performed on disjunctions. For the case of process networks, Türkay and Grossmann (1996) proposed a logic-based Outer-Approximation algorithm. This algorithm consists of solving NLP subproblems in reduced space, in which constraints that do not apply in the disjunctions are disregarded, with which both the efficiency and robustness can be improved. In this method the MILP master problems correspond to the convex hull of the linearization of the nonlinear inequalities. Also, several NLP subproblems must be solved to initialize the master problem in order to cover all the terms in the disjunctions. Penalties can also be added to handle the effect of nonconvexities as in the method by Viswanathan and Grossmann (1990). The above methods have been implemented in the computer prototype LOGMIP, a GAMS-based computer code developed by Vecchietti and Grossmann (1997).

Constraint Programming. This area, which has emerged recently as a logic-based optimization tool, has proved to be particularly successful for certain types of scheduling problems. The basic idea in Constraint Programming (CP) (Van Hentenryck, 1989; Puget, 1994) is to use compact languages for expressing optimization problems in terms of variables that are continuous, integer, and/or Boolean, and constraints that can be expressed in algebraic form (e.g. $h(x) \le 0$), as disjunctions (e.g. $[\, A_1 x \le b_1 \,] \vee [\, A_2 x \le b_2 \,]$), or as conditional logic statements (e.g. If $g(x) \le 0$ then $r(x) \le 0$). In addition the language can support special implicit functions such as the *all different*$(x_1, x_2, ... x_n)$ constraint for assigning different values to the integer variables $x_1, x_2, ... x_n$. The language consists of C++ procedures, although the recent trend has been to provide higher level languages such as OPL. Other commercial CP software packages include ILOG Solver (ILOG, 1999), CHIP (Dincbas *et al*., 1988), ECLiPSe (Wallace *et al*., 1997), and Prolog IV. Rather than relying on traditional mathematical programming methods, CP relies on a tree search using implicit enumeration. The tree is normally enumerated with a depth first search in which the lower bound is given by partial solutions, and the upper bound by the best feasible solution. At each of the nodes in the tree search, constraint propagation is performed through domain reduction of the variables. This involves for instance the reduction of bounds in the case of continuous variables, and/or domains in the case of discrete variables. The former uses procedures for tightening bounds for linear and monotonic functions, while the latter is performed either by inference techniques, or by special procedures. A good example is the "edge-finding" method for jobshop scheduling. Figure 4 presents a simple example of such a method to resolve a disjunction about the relative processing of jobs *i* and *j*.

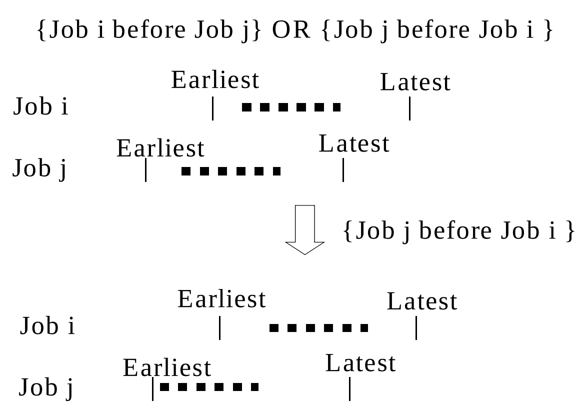

Figure 4. Edge finding technique for jobshop scheduling.

4. A General Disjunctive Model for the Integration of Planning and Scheduling

In the past, planning and scheduling models have largely been solved separately due to the complexity associated with including and solving both levels of decision making in one model. Only very recently have *simultaneous planning and scheduling* models emerged (e.g. Papageorgiou and Pantelides, 1996; Birewar and Grossmann, 1990). While the advances have shown progress towards integration of planning and scheduling, these problems remain in general intractable. This is due to the size of the resulting problem, and the mismatch of the time scales in planning and scheduling. This indicates that there is a need to derive efficient models and algorithms for integrated planning and scheduling. In this section we present a model that reflects the hierarchy of decisions that can be potentially exploited for an efficient solution.

From the review in the previous section, it should be clear that LP and MILP methods, which are extensively used in planning and scheduling, have become quite powerful. In addition, NLP methods are able to tackle increasingly larger problems and are being advanced by rigorous global optimization algorithms. Together, these developments facilitate faster solution of MINLPs. A new exciting direction is logic based optimization methods, such as Generalized Disjunctive Programming and Constraint Programming, which promise to facilitate problem formulation and improve the solution efficiency and robustness. In order to illustrate the use of logic based methods, we present in this section a general GDP model that also has the important feature of integrating planning and scheduling for process networks in a single model (IPS). As will be seen the model gives rise to a generalized disjunctive program that involves embedded disjunctions that reflect the hierarchical nature of decisions involved in the integration problem.

We use a discrete time representation for the planning and the scheduling time domains. Also, we assume that the scheduling model corresponds to the State Task Network (STN) (Kondili *et al.*, 1993). Consider optimizing a given STN superstructure over a time horizon, H. Such a superstructure consists of a set of units, J, capable of performing a set of tasks, I. Feeds, intermediates and products are represented by the set of states, S. In order to integrate both planning and scheduling into the optimization model, H is divided into a number of planning periods, $t = 1..T$, and a number of scheduling periods, $k = 1..K$. The length of a planning period is typically in the order of weeks or a few months, while the length of a scheduling period is typically in the order of hours. We define the set $Int(t,k)$ to denote which of the scheduling periods, k, belong to planning period t. The complete set, parameter and variable definitions are as follows:

Sets:

S	set of states (feeds, intermediates, products)
I	set of tasks
J	set of units
T	set of time periods in the planning horizon
K	set of time periods in the scheduling horizon
$Int(t,k)$	set of scheduling time periods k belonging to planning time period t

Indices:

s	state in set S
i	task in set I
j	unit in set J
t	time period in set T
k	time period in set K

Parameters:

α_{jt}	variable expansion cost for unit j in period t
β_{jt}	fixed expansion cost for unit j in period t
γ_{jt}	fixed operating cost for unit j in period t
c^{p}_{st}	cost associated with state s in planning time period t
c^{s}_{sk}	cost associated with state s in scheduling time period k
c^{r}_{ijk}	cost associated with resource usage for task i on unit j in time period k
η_{ij}	fixed resource cost for task i on unit j
δ_{ij}	variable resource cost for task i on unit j
t^{d}_{ij}	delay time associated with task i on unit j

Variables:
Binary decision variables:
y_j selection of investment in unit *j*
w_{jt} operation of unit *j* in period *t*
z_{jt} capacity expansion of unit *j* in period *t*
v_{ijk} task *i* is performed on unit *j* in period *k*
Continuous decision variables:
Q_{jt} capacity of unit *j* in period *t*
QE_{jt} capacity expansion of unit *j* in period *t*
x_t state variables in period *t*
x^p_{st} subset of state variables for state *s* in planning time period *t*
CO_{jt} operating cost of unit *j* in period *t*
CE_{jt} expansion cost of unit *j* in period *t*
x^s_{sk} subset of state variables for state *s* in scheduling time period *k*
R_{ijk} resource usage for task *i* on unit *j* in time period *k*
B_{ijk} batch size for task *i* on unit *j* in period *k*

Based on the above definitions, the GDP model is as follows:

$$\min Z = \sum_t \left[\sum_j (CO_{jt} + CE_{jt}) + \sum_s c^p_{st} x^p_{st} + \sum_{k \in Int(t,k)} \sum_i \sum_j c^r_{ijk} R_{ijk} + \sum_{k \in Int(t,k)} \sum_s c^s_{sk} x^s_{sk} \right] \quad (1)$$

s.t.

$$g_t(x_t, x_{t-1}) \le a \qquad \forall t \qquad (2)$$

$$f_{sk}\left(x^s_{sk}, x^s_{s,k-1}, x^s_{s,k-t^d}\right) \le b \qquad \forall s,k \qquad (3)$$

$$\begin{bmatrix} y_j \\ \left[\begin{bmatrix} w_{jt} \\ h_{jt}(Q_{jt}, x_t, x_{t-1}) \le d \ (5) \\ CO_{jt} = \gamma_{jt} \ (6) \\ \begin{bmatrix} z_{jt} \\ Q_{jt} = Q_{j,t-1} + QE_{jt} \ (9) \\ CE_{jt} = \alpha_{jt} QE_{jt} + \beta_{jt} (10) \end{bmatrix} \vee \begin{bmatrix} \neg z_{jt} \\ Q_{jt} = Q_{j,t-1} (11) \\ CE_{jt} = 0 \ (12) \end{bmatrix} \\ \begin{bmatrix} v_{ijk} \\ 0 < B_{ijk} \le Q_{jt} \ (13) \\ R_{ijk} = \eta_{ij} + \delta_{ij} B_{ijk} (14) \end{bmatrix} \vee \begin{bmatrix} \neg v_{ijk} \\ B_{ijk} = 0 (15) \\ R_{ijk} = 0 (16) \end{bmatrix} \forall k \in Int(t,k) \end{bmatrix} \vee \begin{bmatrix} \neg w_{jt} \\ D^{jt} x_t = 0 (7) \\ CO_{jt} = 0 (8) \end{bmatrix} \right] \forall t \end{bmatrix} \vee \begin{bmatrix} \neg y_j \\ D^{jt} x_t = 0 (4) \\ \forall t \end{bmatrix} \forall j$$

$$y_j \Rightarrow \bigvee_{t=1}^{T} w_{jt} \qquad \forall j,t \quad (17), \qquad w_{jt} \Rightarrow y_j \qquad \forall j,t \qquad (18)$$

$$w_{jt} \Rightarrow \bigvee_{\tau=1}^{t} z_{j\tau} \qquad \forall j,t \quad (19), \qquad z_{jt} \Rightarrow w_{jt} \qquad \forall j,t \qquad (20)$$

$$w_{jt} \Rightarrow \bigvee_{i,k \in Int(t,k)}^{I} v_{i,j,k} \qquad \forall j,t \quad (21), \qquad v_{i,j,k} \Rightarrow w_{j,t} \qquad \forall i,j,k \in Int(t,k) \qquad (22)$$

$$\Omega_1(y) = True \quad (23), \qquad \Omega_2(v) = True \qquad (24)$$

$CE, CO, Q, QE, x, R, B \ge 0$ (25), $y, w, z, v \in \{True, False\}$ (26)

The objective (1) is to minimize costs over the whole time horizon, and includes operating costs, expansion costs, and costs associated with states over the planning period, as well as resource costs and costs associated with states over the scheduling period. Sales are included by assigning negative values to the appropriate cost coefficients. Global constraints valid for a particular planning period, such as mass balances over mixers, are represented by (2), while global constraints valid for a particular scheduling period, such as inventory constraints, are represented by (3). Note that both (2) and (3) may generally involve 'pass-on' variables from the previous period, giving rise to linking constraints. In addition, the global scheduling constraints (3) may generally also involve a scheduling time delay, t^d, due to processing times, clean-up times, and changeover times.

Constraints (5) – (16) are grouped into a set of nested disjunctive constraints for each unit j. The outer disjunction represents the decision to include unit j in the design or not, which is a strategic planning decision. If unit j is included in the design, (y_j = True), then the set of constraints on the left hand side of the disjunction is applied, otherwise (y_j = False), a subset of state variables associated with unit j are set to zero for all periods through the matrix D^{jt} in (4). The middle disjunction represents the decision to operate unit j in planning period t or not and is only applied if y_j = True. If the unit j operates in period t, (w_{jt} = True), which can be interpreted as either an operational or tactical planning decision, then constraints (5) and (6), as well as the two remaining disjunctions representing expansion and scheduling decisions, are applied. (5) represents constraints that are valid for a given unit j in a particular planning period t, such as unit input-output relationships, while the operating cost of unit j in planning period t is calculated in (6). If unit j does not operate in period t, (w_{jt} = False), a subset of state variables and the operating cost associated with unit j are set to zero for period t through (7) and (8), respectively.

The two inner disjunctions are only applied if w_{jt} = True, and of these the first represents the decision to expand unit j in planning period t or not. If unit j is expanded in period t, (z_{jt} = True), which is also a planning decision, constraints (9) and (10) are applied. (9) states that the capacity at the current period equals the capacity at the previous period plus the capacity expansion, while the expansion cost is calculated in (10). If the decision is not to expand unit j in period t (z_{jt} = False), the capacity remains the same as in the previous period, and the expansion cost is set to zero (see (11) and (12)).

Unit specific scheduling decisions are represented by the second inner disjunction. As pointed out in the previous section there exists no real generalization of scheduling models. We therefore focus on the ideas from STN scheduling first proposed by Kondili *et al*. (1993), since this formulation can be applied to arbitrary network structures. Note that this inner disjunction is only applied for scheduling periods k within the planning period t, as denoted by the set *Int(t,k)*. This disjunction states that if task i is started on unit j in scheduling period k, (v_{ijk} = True), then the batch size is limited by the unit capacity in (13) and the resource usage is calculated in (14). If task i is not started on unit j in period k, (v_{ijk} = False), the starting batch size and resource usage are set to zero in (15) and (16) respectively.

Constraints (17) through (24) are logic propositions representing logical relationships between the discrete variables. (17) states that the inclusion of unit j in the design implies that it must be operated in at least one period t, while (18) states the converse, i.e. that operation of unit j in any period t implies the inclusion of unit j in the design. Similarly, constraint (19) states that operation of unit j in period t implies that it must have been expanded at least once in a previous period, while (20) states the converse that expansion of unit j in period t implies that it will also be operated in that period. Constraint (21) states that the operation of unit j in planning period t implies that at least one task i must be started on unit j in a scheduling period k belonging to planning period t. If a task i starts on unit j in a scheduling period k belonging to t, then unit j must be operated in period t as denoted by (22). Constraint set (23) represents logic propositions relating the discrete design variables, y, for the topology of the network (which combinations are permitted). The relationships among the discrete scheduling variables (v), for example the condition that starting one task on unit j in period k implies that no other task can be started on unit j until task i is finished, are represented by the constraint set (24). Finally, the domains of the variables are given in (25) and (26).

The above represents a conceptual model that integrates planning and scheduling decisions within one single formulation. One advantage of this GDP model is that special structures are revealed, for example the clear hierarchy of decisions from design, operation, and expansion of units to assignment of units and sequencing of tasks. This facilitates the development of tailored algorithms using techniques such as decomposition, as will be discussed in the next section. Furthermore, a GDP model allows the application of specialized logic-based methods that have the effect of reducing non-convexities, and yielding tighter relaxations and ultimately faster solutions. It is also important to note that by fixing some Boolean variables and eliminating subsets of disjunctions, the proposed model can easily shown to reduce to specific forms of planning or scheduling problems.

5. Solution Strategies

While moderately sized planning and scheduling models as presented in sections 2 and 4 can be solved with the mathematical programming methods as discussed in section 3, larger problem instances, which are these days often required for accurate representation of the problem characteristics, require some type of decomposition, aggregation and/or the use of heuristics for their solution. In this section we review some of these approaches that are applicable to large-scale mixed integer linear or nonlinear problems in addition to the methods mentioned in Section 3.

Decomposition

When choosing a decomposition method it is important to consider how to exploit the structure of the model most efficiently and also to choose a degree of decomposition that allows solution in reasonable time while still finding an optimal or near-optimal solution. Several decomposition schemes have been proposed in the literature. Benders decomposition (Benders, 1962) and dual decomposition or Lagrangean relaxation (see e.g. Fisher, 1981) exploit the primal and dual structures of the model respectively. Cross decomposition (see e.g. Van Roy, 1982) exploits both the primal and dual structures and is applicable to models where both the primal and dual subproblems are easy to solve. Bilevel decomposition (e.g. Iyer and Grossmann, 1998b) exploits the structure of models that include different hierarchical levels, such as the hierarchy from design, to planning, to scheduling. Ruszczynski (1997) gives a comprehensive review on decomposition methods for stochastic problems, including cutting plane methods, augmented Lagrangean decomposition, splitting methods and nested decomposition. We discuss only Lagrangean relaxation and bilevel decomposition in further detail below, since an in depth discussion of all decomposition methods is beyond the scope of this paper. The discussion on bilevel decomposition is motivated by its relevance to an example presented in the next section as well as to combined planning and scheduling models such as model (IPS). The discussion on Lagrangean relaxation is motivated by its wide applicability to large-scale optimization models such as (IPS) and its ease of implementation in practice.

Bilevel decomposition. One approach to exploit the hierarchical structure of combined design, planning and/or scheduling models is to decompose the model into an upper level problem at the higher hierarchical level, and a lower level problem at a lower hierarchical level. Iyer and Grossmann (1998b) proposed such a bilevel decomposition algorithm for an MILP design and planning problem, where the upper level involves mainly design decisions while the lower level involves mainly planning decisions. Van den Heever and Grossmann (1999) expanded this approach to MINLPs through the use of GDP. Consider an original model (P) where superscript "d" denotes design variables and superscript "p" denotes planning variables.

$$\begin{array}{ll} \min & f(x^d, y^d, x^p, y^p) \\ s.t. & h(x^d, y^d, x^p, y^p) \leq 0 \qquad \text{(P)} \\ & x \in \Re, y \in \{0,1\} \end{array}$$

To derive the upper level design problem (DP), all the discrete planning variables are relaxed. This results in a much smaller number of nodes in the branch and bound search facilitating a faster solution. Also, some of the constraints and/or variables may be aggregated at this level, indicated by Λ.

$$
\begin{array}{ll}
\min & f(x^d, y^d, x^p, y^p) \\
s.t. & \Lambda h(x^d, y^d, x^p, y^p) \le 0 \\
 & 0 \le y^p \le 1 \\
 & x \in \Re, y^P \in \Re, y^d \in \{0,1\}
\end{array} \quad \text{(DP)}
$$

After (DP) is solved, the discrete design variables are fixed (indicated by the bar on y^d) and the lower level planning problem (PP) is solved for the fixed design.

$$
\begin{array}{ll}
\min & f(x^d, \overline{y}^d, x^p, y^p) \\
s.t. & h(x^d, \overline{y}^d, x^p, y^p) \le 0 \\
 & x \in \Re, y^P \in \{0,1\}
\end{array} \quad \text{(PP)}
$$

Subproblems (DP) and (PP) are solved iteratively and design and integer cuts are added at each iteration to ensure an optimal solution. Note that even though both (DP) and (PP) are in a reduced space, both consider the design and planning model as a whole. A further benefit of this approach is that it significantly reduces the computational effort compared to solving the combined problem as a whole, while still guaranteeing the optimal solution to the original combined model in the convex case. Papageorgiou and Pantelides (1996) proposed a similar decomposition approach for combined campaign planning and scheduling of multipurpose batch/semicontinuous plants. In their work, the upper level problem concerns mainly campaign planning decisions while the scheduling decisions are aggregated, and the lower level problem is solved with some of the campaign planning variables fixed. Again both levels consider the problem as a whole. In the experience of the authors, the bilevel decomposition approach works particularly well for large-scale industrial applications over a long time horizon, especially when combined with the aggregation of time periods as discussed in Example 1.

Lagrangean relaxation. This is an approach that is often applied to models with a block diagonal structure. In such models, distinct blocks of variables and constraints can be identified that are linked with a few "linking" constraints and variables (see Figure 5). Some applications include scenario decomposition for planning under uncertainty (Carøe and Schultz, 1999), unit commitment in power plants (Nowak and Römisch, 1998), midterm production planning (Gupta and Maranas, 1999), oilfield investment planning (Van den Heever *et al*., 2000) and combined transportation and scheduling (Equi *et al*., 1997), to name but a few.

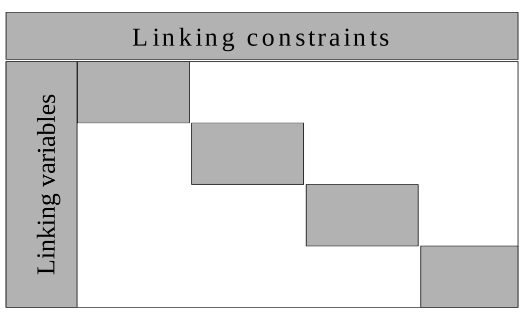

Figure 5. Block diagonal structure

Consider a model (L) that has been partitioned into blocks of constraints $p = 1..P$ where the blocks are linked by a constraint set h:

$$
\begin{array}{ll}
\min & \sum_p f_p(x_p) \\
s.t. & g_p(x_p) \le 0 \quad \forall p \\
 & h(x_1, \ldots, x_p) \le b \\
 & x \in X
\end{array} \quad \text{(L)}
$$

The basic idea behind Lagrangean relaxation as applied to the decomposition of block diagonal structures, is to dualize the linking constraint set, *h*, by removing it and replacing it with a penalty in the objective function involving the associated Lagrangean multipliers, λ, as seen in model (LR):

$$\begin{aligned} \min_x \quad & \sum_p f_p(x_p) + \lambda(h(x_1,\ldots,x_p) - b) \\ s.t. \quad & g_p(x_p) \le 0 \quad \forall p \\ & x \in X \end{aligned} \qquad \text{(LR)}$$

Model (LR) is now decomposable into *P* subproblems and, for any choice of λ, also yields a lower bound to the optimal solution of (L) if the constraints are convex. The case where variables link the blocks can be dealt with by introducing duplicates for each linking variable, setting the duplicates equal, and dualizing this equality constraint. This is referred to as Lagrangean decomposition (Guignard and Kim, 1987). Obtaining the tightest lower bound to (L) requires the solution of the Lagrangean dual problem (LD):

$$\begin{aligned} \max_\lambda \min_x \quad & \sum_p f_p(x_p) + \lambda(h(x_1,\ldots,x_p) - b) \\ s.t. \quad & g_p(x_p) \le 0 \quad \forall p \\ & x \in X \end{aligned} \qquad \text{(LD)}$$

If all the constraints are convex and all the variables are continuous, the optimum of (LD) will equal the optimum of (L). However, a duality gap might exist in the presence of integer variables or other non-convexities, which means that the optimal solution to the dual problem will be strictly less than the true optimum of (L). Guignard (1995) and Bazaara *et al.* (1994) give comprehensive graphical interpretations of the duality gap in the case if integer variables and non-convex constraints respectively. Solving (LD) can be difficult to implement and time consuming, although Fisher (1981) reports on some algorithms for this purpose. A code for solving the dual was developed by Kiwiel (1993/1994), but this code is not widely available to the best of our knowledge. Solving the dual to optimality is therefore often circumvented by using an iterative heuristic approach where (LR) is solved to generate lower bounds to (L) and a heuristic method is used to generate feasible solutions to (L) which are also upper bounds. λ is updated at each iteration with some updating rule, for example a subgradient method (see e.g. Fisher, 1981). This decomposition method reduces the computational effort by solving several subproblems instead of the original problem, and the associated algorithms lend themselves to parallelization to reduce the computational effort even more. For a thorough background on the application of Lagrangean relaxation, we refer the reader to Guignard (1995) and Fisher (1981 and 1985).

Aggregation

For some models, decomposition alone is not enough to obtain a good solution in reasonable time, and some form of aggregation is required to further reduce the model size. Rogers *et al*. (1991) give a good review on the use of aggregation/disaggregation in optimization. These authors define the major components of this framework, namely aggregation analysis, disaggregation analysis and error analysis. The first component involves determining which elements of the model to combine into a single element and how to define the single element, while the second component conversely involves deriving a more refined model from the aggregate one. Error analysis determines the error introduced by aggregation and disaggregation. These three components can be addressed sequentially or iteratively to reduce the computational effort of solving the original problem, with the iterative approach aiming at decreasing the error at each iteration.

It should be noted that the solution to the aggregate formulation is not necessarily feasible for the disaggregate case. However, for certain models it may be possible to formulate the aggregation in such a way as to yield a strict bound to the original problem, and TO guarantee feasibility for the disaggregate level, as shown by Iyer et al. (1998) for the aggregation of oil wells for oil production planning One approach to reduce the number of constraints is to linearly combine some of them into a surrogate constraint where the aggregation coefficients are modified iteratively (see e.g. Ermoliev *et al.*, 1997). Wilkinson *et al.* (1996) use a constraint aggregation approach to solve a large-scale production and distribution planning problem for multiple production sites. In their work an upper level aggregate model is solved to set production targets and also yield a strict upper bound to the original problem, after which the detailed scheduling can be optimized for each site individually with fixed targets thus decreasing the computational effort significantly. Wilkinson (1996) proposed aggregate formulations for large-scale process scheduling problems using ideas of approximation of difference equations, as well as decomposition approaches for solving these models. In the case of multiperiod models, an approach that works well is to aggregate the time periods. This is especially true when the model involves two hierarchical time levels, such as combined design and planning or combined planning and scheduling. Van den Heever and Grossmann (2000) combined the bilevel decomposition approach mentioned above with the aggregation of time periods by aggregating time in the upper level problem with subsequent disaggregation in the lower level planning problem. An additional subproblem is solved after each iteration to determine the best new aggregation scheme (which periods should be grouped together) and information from the aggregation subproblem is used at each iteration to eliminate variables in the lower level problem. It was found that the error introduced by the aggregation of the time periods was very small, mainly due to the optimal aggregation subproblem. Other aggregations schemes include the aggregation of products into families of similar products for the scheduling of multiproduct plants (Kondili *et al*., 1993). Where uncertainty is incorporated through a scenario-based model, scenario aggregation can speed up the solution time significantly. The scenario aggregation approach was applied to a mixed-integer linear multiproduct production planning problem by Jorsten and Leisten (1994) who exploited the coupling between continuous and integer planning variables to allow application of the scenario-aggregation algorithm originally proposed by Rockafeller and Wets (1991) for continuous models.

Apart from decomposition and aggregation techniques, some other heuristic approaches address the solution of large-scale planning and scheduling problems. One such heuristic is a capacity shifting heuristic presented by Ahmed and Sahinidis (2000) for a class of process planning problems. These authors show that the error of their heuristic algorithm vanishes asymptotically as the problem size increases. This is a very nice result, considering that the solution time increases exponentially with the number of time periods for an exact solution algorithm.

6. Examples

In this section we present three examples that illustrate some of the main points covered in this paper. Example 1 deals with a planning problem that gives rise to a large-scale multiperiod MINLP model, and that requires the use of a decomposition/aggregation strategy. Example 2 describes an MILP scheduling model for steel manufacturing that is also tackled through a special decomposition approach. Finally, example 3 describes a hybrid CP/MILP model for a parallel scheduling problem, which demonstrates the advantage of a combined approach as opposed to pure CP or MIP.

Example 1. Hydrocarbon field infrastructure planning

The operation and investment planning involved in the design of hydrocarbon field infrastructures is a challenging problem that involves several complexities such as a long time horizon, nonlinear reservoir behavior, and complex fiscal rules leading to a multiperiod MINLP model with several discrete and continuous variables. In this example (for detail see Van den Heever and Grossmann, 2000) we consider the design, planning and scheduling of an offshore oilfield infrastructure over a planning horizon of 6 years divided into 24 quarterly periods where decisions need to be made. The infrastructure under consideration consists of one Production Platform (PP), 2 Well Platforms (WP), 25 wells and connecting pipelines (see Figure 6). Each oilfield (F) consists of a number of reservoirs (R), while each reservoir in turn contains a number of potential locations for wells (W) to be drilled. Design decisions involve the capacities of the PPs and WPs, as well as decisions regarding which WPs to install over the whole operating horizon. Planning decisions involve the production profiles in each period, as well as decisions regarding when to install PP and WPs included in the design, while scheduling decisions involve the selection and timing of drilling of the wells. This leads to an MINLP model with 9744 constraints, 5953 continuous variables, and 700 0-1 variables.

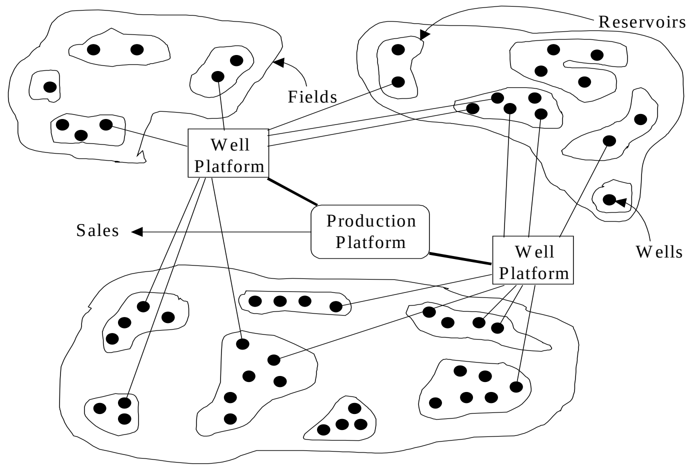

Figure 6: Configuration of fields, well platforms and production platforms

An attempt to solve this model with a commercial package such as GAMS (Brooke and Kendrick, 1992) (using DICOPT (Viswanathan and Grossmann, 1990)) with CPLEX 6.6 (ILOG, 2000) for the MILPs and CONOPT2 (Drud, 1994) for the NLPs on an HP9000/C110 workstation), results in a solution time of 19386 CPU s. To overcome this long solution time, Van den Heever and Grossmann (2000) developed an iterative aggregation/disaggregation algorithm which solved the model in 1423 CPU seconds. This algorithm combines the concepts of bilevel decomposition, time aggregation and logic-based methods. The original design and planning problem is decomposed into an upper level design problem and a lower level planning problem. Both subproblems are formulated as disjunctive models. The upper level design problem is solved in aggregate time, after which a design is fixed, time periods are disaggregated and the lower level planning problems is solved. This result is then used to determine a new time aggregation through a dynamic programming subproblem, integer cuts are added to the design problem, aggregation parameters are updated, and the iteration is repeated until the termination criteria are reached. Thus the application of combined decomposition and aggregation leads to an order of magnitude reduction in solution time, while the same optimal net present value of $ 68 million is found as with DICOPT. For this specific model, the large decrease in computational effort is mainly due to the aggregation/decomposition, while the disjunctive programming formulation contributed mainly towards reducing non-convexities due to zero flows and clarity of representation. However, for different planning models the disjunctive programming approach may reduce the computational effort significantly in addition to the benefits mentioned here, as shown by Van den Heever and Grossmann (1999) for the case of process network design and planning and the retrofit of batch plants.

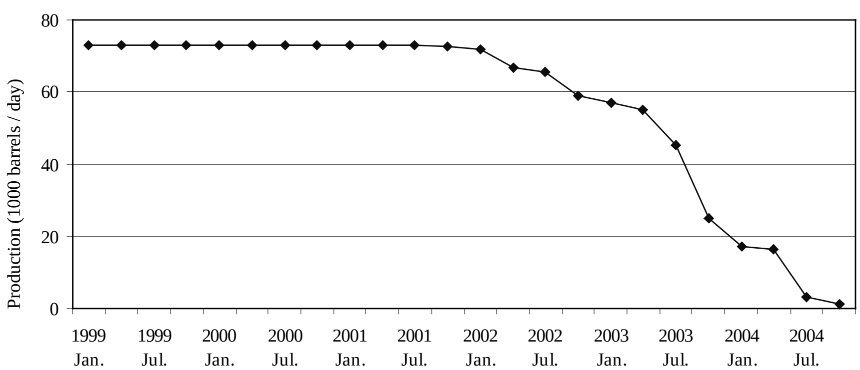

Figure 7: Production profile over 6 year horizon

Figure 7 shows the total oil production over the 6 year horizon, while Table 1 shows the optimal investment plan obtained. Note that only 9 of the 25 wells were chosen in the end. This solution resulted in savings in the order of millions of dollars compared to the heuristic method used in the oilfield industry that specify almost all the wells being drilled.

Table 1. The optimal investment plan

Item		Period invested
PP		Jan. 1999
WP1		Jan. 1999
Reservoir	**Well**	
2	4	Jan. 1999
3	1	Jan. 1999
5	3	Jan. 1999
4	2	Apr. 1999
7	1	Jul. 1999
6	2	Oct. 1999
1	2	Jan. 2000
9	2	Jan. 2000
10	1	Jan. 2000

In Van den Heever *et al.* (2000), the concept of hydrocarbon field infrastructure planning was expanded to include complex fiscal rules such as royalties, tariffs and taxes. This resulted in a model for which no solution could be found by GAMS in more than 5 days. To address this problem, a heuristic solution procedure based on Lagrangean decomposition was proposed that produces several good solutions in a day. This method can potentially be parallelized and combined with the aggregation of time periods to speed up the solution even more.

Example 2. An MILP approach to steel manufacturing

In this section, an MILP approach to produce a production schedule for a steel-making process is discussed. As seen in Figure 8 the steel-making process consists of two furnaces, where the melt steel is combined with scrap and thereafter taken to decarburization and ladle treatment units. Finally the melt steel is solidified in a continuous caster under strictly constrained conditions. There are several complicating factors in the problem such as sequence dependent setup times, maintenance of equipment and production time limitations. Furthermore, temperature and purity issues are critical to the production that also includes low-carbon steel grades. Other problems arising from the metal chemistry as well as plant geometrics are not directly considered in the model. The production steps are illustrated in Figure 8, where the possible paths of the products are shown.

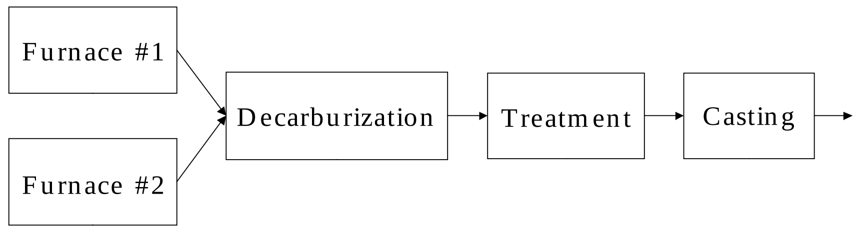

Figure 8: Processing steps in steel manufacturing

A continuous time representation for modeling a large-scale scheduling problem is applied. The proposed decomposition strategy consists of first positioning the orders into blocks, each of which is optimized as a jobshop scheduling problem. Next the blocks are optimally scheduled as a flowshop problem and finally an LP and/or MILP method is used to properly account for setup times and to optimize the allocation of some parallel equipment. This type of approach allows some of the more complicated constraints to be either isolated in a subproblem or inserted as parameters between two solution steps. While this decomposition strategy is not guaranteed to yield the global optimal schedule it allows the solution of very large-scale problems. The strategy is illustrated in Figure 9.

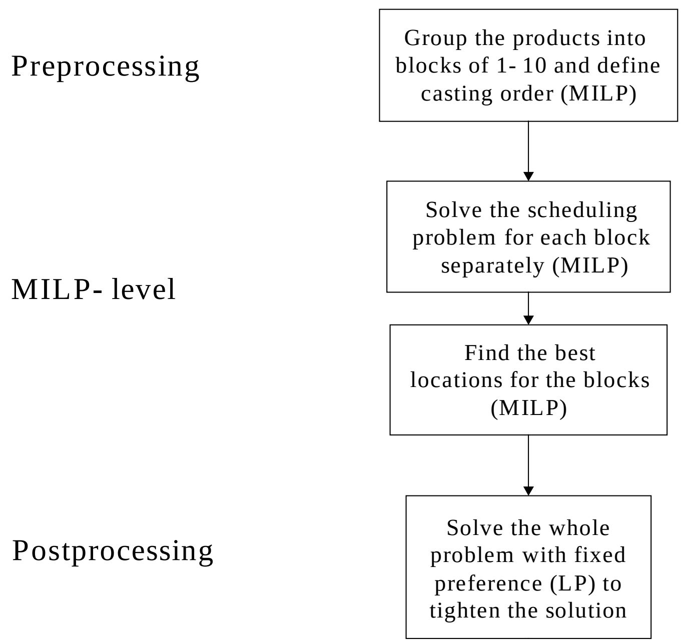

Figure 9: Solution strategy for steel scheduling

The decomposition strategy is motivated by the fact that heats with similar product properties can be grouped into sequences in a preprocessing step. These sequences are then treated as independent blocks by first optimizing their internal production order and thereafter finding the optimal sequence between the blocks. In a postprocessing step, the heats are again treated as individual products with fixed ordering and the gaps, caused by the grouping, are closed by solving an LP-problem. The solution can furthermore be improved by solving a final MILP. This example forms an interesting approach where a problem is both decomposed and rejoined through optimization and thus one large MILP is replaced by a number of smaller and solvable subproblems. Also, the modularity of the procedure makes it possible to solve only parts of the problem when changes occur. Table 2 shows the results that were obtained from solving a one week problem containing 81 products. The optimization was performed on a Linux-platform using XPRESS-MP in GAMS.

Table 2: Comparison of results between proposed approached and conventional jobshop model

Problem	Products	Groups	CPU-s	Makespan	Integer Gap
Proposed decomposition	81	19	1035.3	132h 6 min	2%
Conventional jobshop	10	1	>10,000	19h 7min	52%

The 81 product problem is not solvable with standard MILP methods, as seen from the failure in solving even the 10-product jobshop-example in reasonable time (the makespan is the best one obtained at 10,000 CPU s.). The proposed strategy solves the complete problem in less than 20 CPU minutes. Even though the decomposition strategy is not expected to result into a global optimal solution, the maximum deviation of the makespan with respect to a theoretical optimum is only 2% and the makespan is reduced from one week to 5 days and 12 hours.

Example 3. A hybrid MILP/CP approach for parallel scheduling

In this example, a strategy of decomposing a scheduling problem into a CP and MILP part is discussed. The basic idea is to combine the two methods such that their complementary strengths can be exploited. The problem is a single stage scheduling problem with parallel units reported by Jain and Grossmann (2000). In the decomposition strategy a relaxed MILP is solved at the master level and a feasibility subproblem is solved with CP. The relaxed MILP excludes the complicating constraints which in this case are the sequencing inequalities. Those are reformulated as a feasibility CP subproblem. The strategy consists of two main steps: the relaxed MILP is first solved to its global optimal solution to obtain a feasible assignment and thereafter a feasibility check is performed by solving a CP sequencing problem with fixed assignment (see Figure 10).

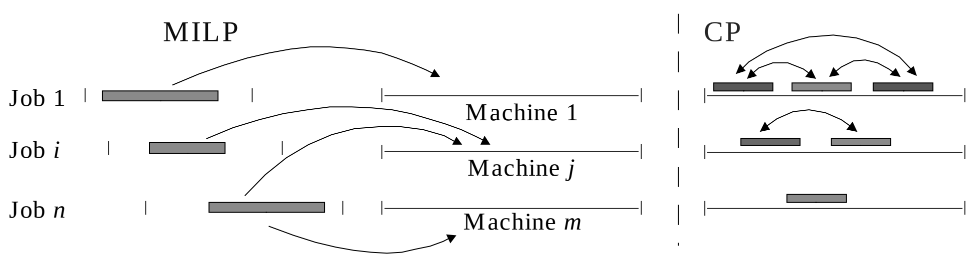

Figure 10: Scheduling of parallel machines

If the sequencing problem is infeasible, cuts are added to the next relaxed MILP to exclude the previous infeasible assignment. The procedure continues solving alternate CP and MILP problems until a feasible sequence is found. The communication between the solution steps is done through fixing assignment variables and generating integer cuts.

This strategy requires that the problem be decomposed into two subproblems of which the MILP (the assignment problem) provides a tight LP-relaxation and contains the objective function variables, and the CP (the sequencing problem) has no objective function variables and includes the constraints with poor relaxations. This ensures an efficient assignment of machines and that the first feasible sequence found is optimal. As can be seen in the following mathematical formulation, equivalence relations are also needed to join the variables in the two subproblems due to some structural differences. Here we will only present the main elements of the hybrid formulation. For more details, we refer to Jain and Grossmann (2000).

$$\min \sum_{i \in I} \sum_{m \in M} C_{im} x_{im} \tag{1}$$

s.t.

$$ts_i \geq r_i \quad \forall i \in I \tag{2}$$

$$ts_i \leq d_i - \sum_{m \in M} p_{im} x_{im} \quad \forall i \in I \tag{3}$$

$$\sum_{m \in M} x_{im} = 1 \quad \forall i \in I \tag{4}$$

$$\sum_{i \in I} x_{im} p_{im} \leq \max_i \{d_i\} - \min_i \{r_i\} \quad \forall m \in M \tag{5}$$

$$\text{if } (x_{im} = 1) \text{ then } (z_i = m) \quad \forall i \in I, m \in M \tag{6}$$

$$i.start \geq r_i \quad \forall i \in I \tag{7}$$

$$i.start \leq d_i - p_{z_i} \quad \forall i \in I \tag{8}$$

$$i.duration = p_{z_i} \quad \forall i \in I \tag{9}$$

$$i \text{ requires } t_{z_i} \quad \forall i \in I \tag{10}$$

$$ts_i \geq 0 \tag{11}$$

$$x_{im} \in \{0,1\} \quad \forall i \in I, m \in M \tag{12}$$

$$z_i \in M \quad \forall i \in I \tag{13}$$

$$i.start \in Z \quad \forall i \in I \tag{14}$$

$$i.duration \in Z \quad \forall i \in I \tag{15}$$

$$\sum_{i \in I} a_{im}^j x_{im} \leq \sum_{i \in I} a_{im}^j - 1 \quad \forall m \in M \tag{16}$$

The assignment MILP problem is defined by Eqs. (1-5). The objective function (1) minimizes the processing costs for all jobs. The binary variable, x_{im} , equals one if job *i* is assigned to machine *m*, else it is zero. Constraint (2) and (3) ensure that processing of a job *i* starts after the release date and is completed before the duedate. Each job needs exactly one machine as is stated in Eq. (4) and the last MILP constraint (5) tightens the LP-relaxation. After solving the MILP problem the fixed assigments are transferred to the CP model using Eq. (6).

The sequencing CP problem, given in Eqs. (7-10), is then solved separately for each machine. In the formulation, *i* is an activity (job), the start time of which is specified in Eqs. (7) and (8) and duration in Eq. (9). It should be noted that it is possible to use a variable as an index in CP and p_{zi} refers to the processing time of job *i* in the assigned equipment. In constraint (10) the special construct *requires* enforces that job *i* needs a unary recourse from the set of resources *t*. If a machine cannot be scheduled, a cut of the form (16) is added to the next assignment MILP problem.

In Table 3, the problems are solved using complete MILP and CP formulations, as well as the hybrid model, with modified data given in Harjunkoski et al. (2000) where the original release dates, due dates and durations (Jain and Grossmann, 2000) have been arbitrarily changed and roughly multiplied by a factor of 10 to test the robustness of the method. The problems were solved on a Sun workstation with OPL Studio (ILOG, 1999) using CPLEX 6.5 and ILOG Solver (ILOG, 1999) and Scheduler (ILOG, 1999). In the following table the CPU times are listed for the four test problems.

Table 3: Computational Results in Parallel Scheduling Problem

Problem	1	2	3	4
Machines/Jobs	3/7	3/12	5/15	5/20
MILP	0.58	164.92	528.86	>40,000
CP	0.04	3.35	590.9	11666.4
HYBRID	0.49	5.27	0.56	35.64

In the hybrid formulation, most of the CPU time is consumed by the MILP. Even though CP overperforms the hybrid approach in the smallest problems the results clearly show that both CP and MILP suffer from combinatorial explosion, but the combination of these two methods performs very well even for the largest problem. It should be pointed out that the hybrid strategy does not compromise the global optimality.

7. Conclusions

This paper has presented an overview of planning and scheduling. It has been shown that these problems lead to discrete optimization models for which the associated mathematical programming problems correspond to integer programming problems, which can exhibit exponential behavior in their computation. Logic-based optimization techniques offer the potential of not only simplifying the formulations, but also of decreasing the computational requirements. We have illustrated the use of logic based optimization as a modeling tool through a novel Generalized Disjunctive Program, that integrates planning and scheduling for process networks, and where the scheduling is represented with the STN model. We have also given a brief overview of decomposition strategies since these are essential in tackling large scale industrial problems. Finally, we have presented three examples, planning of oilfields, scheduling of steel manufacturing and scheduling of parallel machines, to illustrate the application of new techniques that are making possible the solution of problems that were essentially unsolvable a few years ago.

While the integration of planning and scheduling remains a major challenge due to the potentially large size of the resulting optimization problem, another major challenge that has not been covered in this paper is the integration of planning and scheduling with control. This is essentially virgin territory in which very little work has been reported. The reader is referred for instance to the work by Bose and Pekny (2000), Perea *et al.* (2000), and Vargas and Rivera (2000) who have addressed the incorporation of model predictive control in scheduling, and the dynamics and control of supply chains. At present, however, these works have addressed only specific applications due to the lack of a general framework, which possibly might be achieved through the use of hybrid systems (Kowalewski, 2000; Morari, 2000).

References

Ahmed, S. & Sahinidis, N.V. (1998). Robust process planning under uncertainty. *Ind. Eng. Chem. Res., 37,* 1883-1892.

Ahmed, S. & Sahinidis, N.V. (2000). Analytical investigations of the process planning problem. *Computers and Chemical Engineering, 23*, 1605-1621.

Applequist, G. J.F. Pekny, and G.V. Reklaitis (2000), "Risk and Uncertainty in Managing Chemical Manufacturing Supply Chains," *Computers and Chemical Engineering*, Vol. 24, no. 9-10, p. 2211-2222.

Balas, E. (1985). Disjunctive Programming and a hierarchy of relaxations for discrete optimization problems. *SIAM J. Alg. Disc. Meth., 6*, 466-486.

Balas, E., Ceria, S. & Cornuejols, G.A. (1993). Lift-and-Project Cutting Plane Algorithm for Mixed 0-1 Programs. *Mathematical Programming, 58,* 295-324

Bazaraa, M.S., Sherali H.D. & Shetty C.M. (1994). Nonlinear Programming, John Wiley.

Beaumont, N. (1991). An Algorithm for Disjunctive Programs. *European Journal of Operational Research, 48*, 362-371.

Benders, J.F. (1962). Partitioning Procedures for Solving Mixed-variables Programming Problems. *Numerische Mathematik, 4*, 238-252.

Birewar, D.B. & Grossmann, I.E. (1990). Simultaneous Production Planning and Scheduling of Multiproduct Batch Plants. *Ind. Eng. Chem. Res., 29*, 570.

Bok, J-K., Grossmann, I.E. & Park, S. (2000). Supply chain optimization in continuous flexible process networks. *Ind. Eng. Chem. Res., 39*, 1279-1290.

Borchers, B. & Mitchell, J.E. (1994). An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programming. *Computers & Operations Research, 21*, 359-367.

Bose, S. & Pekny, J.F. (2000). A model predictive framework for planning and scheduling problems: a case study of consumer goods supply chain. *Computers & Chemical Engineering, 24*, 329-335.

Brooke, A., Kendrick, D. & Meeraus, A., (1992). GAMS - A User's Guide. *Scientific Press, Palo Alto*.

Carøe, C.C. & Schultz, R. (1999). Dual Decomposition in Stochastic Integer Programming., *Operations Research Letters, 24 (1-2)*, 37.

Chvatal (1983). Linear Programming. *Freeman*.

Clay, R.L. & Grossmann, I.E. (1997). A Disaggregation algorithm for the optimization of stochastic planning models. *Computers & Chemical Engineering, 21*, 751.

Crainic, T.G. & Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research, 97*, 409-438.

Dash Associates (1999). XPRESS-MP, User Guide.

Dimitriadis, A.D., Shah, N. & Pantelides, C.C. (1997). RTN-Based Rolling Horizon Algorithms for Medium Term Scheduling of Multipurpose Plants. *Computers & Chemical Engineering, 21*, S1061.

Dincbas, M., Van Hentenryck, P., Simonis, H., Aggoun, A., Graf, T. & Berthier, F. (1988). The constraint logic programming language CHIP. *In FGCS-88: Proceedings of International Conference on Fifth Generation Computer Systems, Tokyo*, 693-702.

Ding-Mai & Sargent R.W.H. (1992). A Combined SQP and Branch and Bound Algorithm for MINLP Optimization. *Internal Report, Centre for Process Systems Engineering, London*

Drud, A. (1992). CONOPT – A Large Scale GRG Code. *ORSA Journal on Computing 6*, 207.

Duran, M.A. & Grossmann I.E. (1986). An Outer-Approximation Algorithm for a Class of Mixed-integer Nonlinear Programs. *Math Programming, 36*, 307.

Erengüç, Ş.S., Simpson, N.C. & Vakharia, A.J. (1999). Integrated production/distribution planning in supply chains: An invited review. *European Journal of Operational Research, 115*, 219-236.

Equi, L., Gallo, G., Marziale, S. & Weintraub, A. (1997). A combined transportation and scheduling problem. *European Journal of Operational Research, 97*, 94-104.

Ermoliev, Y.M., Kryazhimskii, A.V. & Ruszczynski, A. (1997). Constraint aggregation principle in convex optimization. *Mathematical Programming, 76*, 353-372.

Fisher M.L. (1985). An applications oriented guide to lagrangean relaxation. *Interfaces, 15*, 10-21.

Fisher, M.L. (1981). The Lagrangean relaxation method for solving integer programming problems. *Management Science, 27 (1)*, 1–17.

Fletcher, R. (1987), Practical Methods of Optimization, *Wiley*.

Fletcher, R. & Leyffer, S. (1994). Solving Mixed Integer Nonlinear Programs by Outer Approximation. *Math Programming 66*, 327.

Floudas, C.A. (2000). Global Optimization in Design and Control of Chemical Process Systems. *Preprints IFAC Symposium DYCOPS 5, Korfu*, 167-176.

Floudas, C.A., Adjiman, C.S., Androulakis, I.P. & Maranas, C.D. (1996). A Global Optimization Method, αBB, for Process Design. *Computers & Chemical Engineering, 20*, S419.

Fourer, R., Gay, D.M. & Kernighan, B.W. (1992). AMPL: A Modeling Language for Mathematical Programming. *Duxbury Press, Belomont, CA*.

Geoffrion, A. M. (1972). Generalized Benders Decomposition. *Journal of Optimization Theory and*

Applications, 10 (4), 237-260.

Goldberg, D.E. (1989). Genetic Algorithms in Search, Optimisation and Machine Learning. *Addison-Wesley, Reading Mass.*

Grossmann, I.E. (ed.) (1996), Global Optimization in Engineering Design, *Kluwer, Dordrecht.*

Grossmann, I.E. & Kravanja, Z. (1997). Mixed-integer Nonlinear Programming: A Survey of Algorithms and Applications. *The IMA Volumes in Mathematics and its Applications, 93, Large-Scale Optimization with Applications. Part II: Optimal Design and Control (eds, Biegler, Coleman, Conn, Santosa), Springer Verlag,* 73-100.

Grossmann, I.E., Quesada, J., Raman, R. & Voudouris, V. (1996). Mixed Integer Optimization Techniques for the Design and Scheduling of Batch Processes. *Batch Processing Systems Engineering (Eds. G.V. Reklaitis, A.K. Sunol, D.W.T. Rippin, O. Hortacsu), Springer-Verlag, Berlin*, 451-494.

Guignard, M. (1995) Lagrangean relaxation: A short course. *Belgian Journal of OR: Special Issue Francoro, 35*, 3.

Guignard, M. & Kim, S. (1987). Lagrangean decomposition: A model yielding stronger lagrangean bounds. *Mathematical programming, 39*, 215-228.

Gupta, A. & Maranas, C.D. (1999). A Hierarchical Lagrangean Relaxation Procedure for Solving Midterm Planning Problems. *Ind. Eng. Chem. Res., 38*, 1937-1947.

Gupta, O.K. & Ravindran, V. (1985). Branch and Bound Experiments in Convex Nonlinear Integer Programming. *Management Science, 31 (12)*, 1533-1546.

Han, S.P. (1976). Superlinearly Convergent Variable Metric Algorithms for General Nonlinear Programming Problems. *Math Programming, 11*, 263-282.

Harjunkoski, I., Jain, V. & Grossmann, I.E. (2000). Hybrid Mixed-Integer/Constraint Logic Programming Strategies for Solving Scheduling and Combinatorial Optimization Problems, *Computers & Chemical Engineering, 24*, 337-343.

Hobbs, B.F. (1995). Optimization methods for electric utility resources planning. *European Journal of Operational Research, 83*, 1-20.

Hooker, J. N. & Osorio, M. A. (1997). Mixed logic/linear programming. *Discrete Applied Mathematics, to appear*.

Horst, R. & Pardalos, P.M. (eds.) (1995). Handbook of Global Optimization. *Kluwer*.

Horst, R. & Tuy, H. (1993). Global Optimization. *Springer-Verlag*.

IBM (1992). IBM Optimization Subroutine Library Guide and Reference. *IBM Systems Journal SC23-0519, 31 (1).*

Ierapetritou, M. & Floudas, C.A. (1998). Effective continuous-time formulation for short-term scheduling. *I&EC Res., 37*, 4341-4359.

Ierapetritou, M.G., Avecedo, J. & Pistokopoulos, E.N. (1996). An optimization approach for process engineering problems under uncertainty. *Computers & Chemical Engineering, 20 (6/7)*, 703-709.

ILOG (1999). ILOG OPL Studio 2.1., User's Manual. *ILOG Inc*.

ILOG (1999). ILOG Scheduler 4.4., User's Manual. *ILOG Inc.*

ILOG (1999). ILOG Solver 4.4., User's Manual. *ILOG Inc*.

ILOG (2000). ILOG CPLEX 6.6., User's Manual. *ILOG Inc*.

Iyer, R.R. & Grossmann, I.E., (1998a). Synthesis and operational planning of utility systems for multiperiod operation. *Computers & Chemical Engineering, 22*, 979-993.

Iyer, R.R. & Grossmann, I.E. (1998b). A Bilevel Decomposition Algorithm for Longe-Range planning of process networks. *Ind. Eng. Chem. Res., 37*, 474.

Iyer, R.R., Grossmann, I.E., Vasantharajan, S. & Cullick, A.S. (1998). Optimal planning and scheduling of offshore oil field infrastructure investment and operations. *Ind. Eng. Chem. Res.*, *Vol. 37*, 1380.

Jain, V. & Grossmann, I.E. (2000), Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems. *presented at INFORMS, Paper SD32.1, Salt Lake City*.

Jorsten, K. & Leisten, R. (1994). Scenario aggregation in single-resource production planning models with uncertain demand. *Production Planning and Control, 5 (3)*, 271-281.

Kirkpatrick, S., Gelatt, C.D. & Vechi, M.P. (1983). Optimization by Simulated Annealing. *Science, 220*, 671.

Kiwiel, K.C. (1993/1994). User's Guide for NOA 2.0/3.0: A Fortran package for convex nondifferentiable optimization. *Polish Academy of Sciences, Systems Research Institute, Warsaw, Poland.*

Kocis, G.R. & Grossmann, I.E. (1987). Relaxation Strategy for the Structural Optimization of Process Flow Sheets. *Ind. Eng. Chem. Res. 26*, 1869.

Kondili E. Pantelides C.C. & Sargent R.W.H. (1993). A General Algorithm for Short-Term Scheduling of Batch Operations-I. MILP Formulation. *Computers & Chemical Engineering, 17 (2)*, 211-227.

Kowalewski, S. (2000). Hybrid Systems in Process Control: Challenges, Methods, and Limits. *CPC-6*.

Ku, H., Rajagopalan, D. & Karimi, I.A. (1987). Scheduling in Batch Processes. *Chem. Engng. Prog.*, 35-45.

Lee, S. & Grossmann, I.E. (1999). Nonlinear Convex Hull Reformulations and Algorithms for Generalized Disjunctive Programming. *submitted for publication*.

Liu, M.L. & Sahinidis, N.V. (1996). Optimization in Process planning under uncertainty. *Ind. Eng. Chem. Res., 35*, 4154-4165.

Marsten, R., Saltzman M., Lustig, J. & Shanno, D. (1990). Interior Point Methods for Linear Programming: Just Call Newton, Lagrange and Fiacco and McCormick! *Interfaces*, *20(4)*, 105-116.

Mazzola, J.B. & Neebe, A.W. (1999). Lagrangian relaxation based solution procedures for a multiproduct capacitated facility location problem with choice of facility type. *European Journal of Operational Research, 115*, 285-299.

McDonald, C. & Karimi, I.A. (1997). Planning and Scheduling of Parallel Semicontinuous Processes. 1. Production Planning. *Ind. Eng. Chem. Res., Vol. 36*, 2691.

Minoux, M. (1983). Mathematical Programming: Theory and Algorithms, *John Wiley*.

Mockus, L. & Reklaitis, G.V. (1996). Continuous time representation in batch/semi-continuous processes: randomized heuristic approach. *Computers & Chemical Engineering*, *20*, S1173-S1177.

Morari, M. (2000). Hybrid System Analysis and Control via Mixed Integer Optimization. *CPC-6*.

Moro, L.F.L, Zanin, A.C. & Pinto, J.M. (1998). A planning model for refinery diesel production., *Computers & Chemical Engineering*, *22,* S1039.

Murtagh, B.A. & Saunders, M.A. (1978). Large-Scale Linearly Constrained Optimization. *Mathematical Programming, 14*, 41-72.

Murtagh, B.A. & Saunders, M.A. (1982). A Projected Lagrangian Algorithm and its Implementation for Sparse Nonlinear Constraints, *Mathematical Programming Study, 16,* 84-117.

Nemhauser, G. L. & Wolsey, L. A. (1988). Integer and Combinatorial Optimization. *Wiley-Interscience, New York*.

Nowak, M.P. & Römisch, W. (1998). Stochastic Lagrangean Relaxation applied to Power Scheduling in a Hydro-Thermal System under Uncertainty. *Institut für Mathematik Humboldt-Universität Berlin, Preprint*.

Owen, S.H. & Daskin, M. S. (1998). Strategic facility location: A review. *European Journal of Operational*

Research, 111, 423-447.

Pantelides, C.C. (1994). Unified frameworks for optimal process planning and scheduling. *In Foundations of Computer Aided Process Operations, Rippin, D.W.T., Hale J.C. and Davis, J.F. (Eds.), CACHE, Austin, TX*, 253-274.

Papageorgiou, L.G. & Pantelides, C.C. (1996a). Optimal campaign planning/scheduling of multipurpose batch/semi-continuous plants, 1. Mathematical formulation. *Ind. Eng. Chem. Res., 35*, 488.

Papageorgiou, L.G. & Pantelides C.C. (1996b). Optimal campaign planning/scheduling of multipurpose batch/semi-continuous plants, 2. A mathematical decomposition approach. *Ind. Eng. Chem. Res., 35*, 510.

Pekny, J.F. & Reklaitis, G.V. (1998). Towards the Convergence of Theory and Practice: A Technology Guide for Scheduling/Planning Methodology. *AIChE Symposium Series, 94 (320)*, 91-111

Pekny, J.F. & Zentner, M.G. (1994). Learning to solve process scheduling problems: the role of rigorous knowledge acquisition frameworks. *In Foundations of Computer Aided Process Operations, Rippin, D.W.T., Hale J.C., Davis, J.F. (Eds.), CACHE, Austin (TX)*, 275-309.

Perea, E., Grossmann, I.E. & Ydstie, E. (2000). Towards the integration of dynamics and control for supply chain management. *Computers & chemical engineering, 24*, 1143-1150.

Pinto, J. & Grossmann, I.E. (1998). Assignment and Sequencing Models for the Scheduling of Chemical Processes. *Annals of Operations Research, 81*, 433-466.

Powell, M.J.D. (1978). A Fast Algorithm for Nonlinearly Constrained Optimization Calculations. *In Numerical Analysis, Dundee 1977. G.A. Watson (ed.), Lecture Notes in Mathematics 630, Springer-Verlag, Berlin*

Puget, J.-F. (1994). A C++ Implementation of CLP. *Technical Report, Ilog Solver Collected Papers, Ilog, France.*

Quesada, I. & Grossmann, I.E. (1992). An LP/NLP Based Branch and Bound Algorithm for Convex MINLP Optimization Problems. *Computers & Chemical Engineering, 16*, 937-947

Raman, R. & Grossmann, I.E. (1991). Relation Between MILP Modelling and Logical Inference for Chemical Process Synthesis. *Computers & Chemical Engineering, 15*, 73

Raman, R. & Grossmann, I.E. (1994). Modelling and Computational Techniques for Logic Based Integer Programming. *Computers & Chemical Engineering, 18*, 563

Reklaitis, G.V. (1991) Perspectives on scheduling and planning of process operations. *Presented at the Fourth international symposium on process systems engineering, Montebello (Canada).*

Reklaitis, G.V. (1992) Overview of scheduling and planning of batch process operations, *NATO Advanced Study Institute- Batch Process Systems Engineering, Antalya (Turkey).*

Rippin, D.W.T. (1993) Batch process systems engineering: a retrospective and prospective review. *Computers & Chemical Engineering, 17*, S1-S13.

Rockafeller, R.T. & Wets, R.J.-B. (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research, 16 (1)*, 119-147.

Rogers, D.F., Plante, R.D., Wong, R.T. and Evans, J.R. (1991). Aggregation and disaggregation techniques and methodology in optimization. *Oper. Res., 39*, 553.

Ruszczynski, A. (1997). Decomposition methods in stochastic programming. *Mathematical Programming, 79*, 333-353.

Sahinidis, N.V., Grossmann, I.E., Fornari, R.E. & Chathrathi, M. (1989). Optimization Model for Long Range Planning in the Chemical Industry. *Computers & Chemical Engineering, 13 (9)*, 1049-1063.

Sahinids, N.V. & Ryoo, H.S. (1995). Global Optimization of Nonconvex NLP's and MINLP's with Applications in Process Design, *Computers & Chemical Engineering, 19*, 551

Saigal (1995). Linear Programming: A Modern Integrated Analysis. *Kluwer Academic Publishers.*

Sand, G., Engell, S., Markert, A., Schultz, R. & Schulz, C. (2000). Approximation of an ideal online scheduler for a multiproduct batch plant. *Computers & chemical engineering, 24 (2-7)*, 361-367.

Schittkowski, K. (1981). The Nonlinear Programming Method of Wilson, Han and Powell with an Augmented Lagrangian Type Line Search Function. Part 1: Convergence Analysis. *Numerische Mathematic*, *38*, 83-114. Part 2: An Efficient Implementation with Linear Least Squares Subproblems. *op. cit.*, 115-127.

Shah, N. (1998). Single- and multisite planning and scheduling: Current status and future challenges. *AIChE Symposium Series 94 (320)*, 75.

Shah, N. & Pantelides, C.C. (1991). Optimal Long-Term Campaign Planning and Design of Batch Operations. *Ind. Eng. Chem. Res., 30*, 2308-2321.

Shah, N., Pantelides, C.C. & Sargent, R.W.H. (1993). A general algorithm for short-term scheduling of batch operations. II. Computational issues. *Computers & chemical engineering, 17 (2),* 229-244.

Smith, E. & Pantelides, C. (1996). Global Optimisation of General Process Models, *in Grossmann, I.E. (ed.), Global Optimization in Engineering Design, Kluwer, Dordrecht*, 355-386.

Stubbs, R.A. & Mehrotra. S. (1996). A Branch and Cut Method for 0-1 Mixed Convex Programming, *presented at INFORMS Meeting, Washington*.

Türkay, M. & Grossmann, I.E. (1996). A Logic Based Outer-Approximation Algorithm for MINLP Optimization of Process Flowsheets, *Computers & Chemical Engineering, 20*, 959-978

Tawarmalani, M. & Sahinidis, N.V. (2000). Global Optimization of Mixed Integer Nonlinear Programs: A Theoretical and Computational Study. *submitted for publication*.

Van den Heever, S.A. & Grossmann, I.E. (1999). Disjunctive multiperiod optimization methods for design and planning of chemical process systems. *Computers & Chemical Engineering, 23*, 1075-1095.

Van den Heever, S.A. & Grossmann, I.E. (2000). An Iterative Aggregation/Disaggregation Approach for the Solution of a Mixed-Integer Nonlinear Oilfield Infrastructure Planning Model. *Ind. Eng. Chem. Res., 39 (6)*, 1955-1971.

Van den Heever, S.A., Grossmann, I.E., Vasantharajan, S. & Edwards, K. (2000). A Lagrangean Decomposition Approach for the Design and Planning of Offshore Hydrocarbon Field Infrastructures with Complex Economic Objectives. *Submitted for publication, August 2000*.

Van Hentenryck, P. (1989). Constraint Satisfaction in Logic Programming. *MIT press, Cambridge, MA.*

Van Roy, T.J. (1983). Cross Decomposition for Mixed Integer Programming. *Mathematical Programming, 25*, 46-63.

Vargas-Villamil, F.D. & Rivera, D.E. (2000). Multilayer optimization and scheduling using model predictive control: application to reentrant semiconductor manufacturing lines. *Computers & Chemical Engineering, 24*, 2009-2021.

Vasantharajan, S., Viswanathan, J. & Biegler, L.T. (1990). Reduced Successive Quadratic Programming implementation for large-scale optimization problems with smaller degrees of freedom. *Computers & Chemical Engineering, 14*, 907.

Vecchietti, A. & Grossmann, I.E. (1997). LOGMIP: A Discrete Continuous Nonlinear Optimizer, *Computers & Chemical Engineering, 21*, S427-S432

Viswanathan, J. & Grossmann, I.E. (1990). A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization. *Computers & Chemical Engineering, 14*, 769.

Wallace, M., Novello, S. & Schimpf, J. (1997). ECLiPSe: a Platform for Constraint Logic Programming, *ICL Systems Journal, 12 (1)*, 159-200.

Westerlund, T. & Pettersson, F. (1995). A Cutting Plane Method for Solving Convex MINLP Problems, *Computers & Chemical Engineering*, *19*, S131-S136.

Wilkinson, S.J. (1996). Aggregate formulations for large-scale process scheduling problems. *Thesis dissertation, department of chemical engineering and chemical technology, Imperial College, London.*

Wilkinson, S.J., Cortier, A., Shah, N. & Pantelides, C.C. (1996). Integrated production and distribution scheduling on a Europe-wide basis. *Computers & Chemical Engineering*, *20*, S1275-S1280.

Yuan, X., Zhang, S., Piboleau, L. & Domenech, S. (1988). Une Methode d'optimisation Nonlineare en Variables pour la Conception de Procedes, *Operations Research*, *22*, 331.

Zhang, X. & Sargent, R.W.H. (1996). The optimal operation of mixed-production facilities-a general formulation and some approaches for the solution., *Computers & Chemical Engineering, 20*, S897-S904.