0263–8762/00/$10.00+0.00
© Institution of Chemical Engineers
Trans IChemE, Vol 78, Part A, September 2000

MIXED INTEGER OPTIMIZATION IN THE CHEMICAL PROCESS INDUSTRY

Experience, Potential and Future Perspectives

J. KALLRATH

BASF-AG, ZDP/C (Scientific Computing), Ludwigshafen, Germany

Proper organization, planning and design of production, storage locations, transportation and scheduling are vital to retain the competitive edge of companies in the global economy. Typical additional problems in the chemical industry suitable for optimization are process design, process synthesis and multi-component blended-flow problems leading to nonlinear or even mixed integer nonlinear models. Mixed integer optimization (MIP) determines optimal solutions of such complex problems; the development of new algorithms, software and hardware allow the solution of larger problems in acceptable times.

This tutorial paper addresses two groups. The focus towards the first group (managers and senior executives) and readers who are not very familiar with mathematical programming is to create some awareness regarding the potential benefits of MIP, to transmit a sense of what kind of problems can be tackled, and to increase the acceptance of MIP. The second group (a more technical audience with some background in mathematical optimization), might rather appreciate the state-of-the-art view on good-modelling practice, algorithms and an outlook into global optimization.

Some real-world MIP problems solved by BASF's mathematical consultants are briefly described, among them discrete blending and multi-stage production planning and distribution with several sites, products and periods. Finally, there is a focus on future perspectives and sources of MIP support are indicated from academia, software providers and consulting firms.

Keywords: modelling; production planning; scheduling; pooling problem; mixed integer linear and nonlinear programming; Branch & Bound; Branch & Cut; LP-relaxation; Outer-Approximation; global optimization

1 INTRODUCTION

What is optimization and what are optimization problems? In an optimization problem (OP), one tries to minimize or maximize a global characteristic of a decision process such as elapsed time or cost, by exploiting certain available degrees of freedom under a set of restrictions (constraints). OPs arise in almost all branches of industry, e.g., in product and process design, production, logistics and even strategic planning. While the word optimization, in nontechnical language, is often used in the sense of improving, the mathematical optimization community sticks to the original meaning of the word related to finding the best solution either globally or at least in a local neighbourhood. Except for very simple cases, OPs cannot be solved by simulation [also called parameter studies], i.e., by simulating the processes under investigation, evaluating the objective function and comparing the results. Since experts of simulation techniques in charge of these OPs have developed intuition and heuristics to select appropriate scenarios to be evaluated, and simulation software exists to perform their evaluation, simulation may lead to reasonable results, but there is no guarantee that the optimal solution or even a solution close to the optimum is found. This is especially troublesome for complex problems, or those which require decisions with large financial impact.

What is required when it is necessary to solve a real world problem by mathematical optimization? The first thing needed is to represent the problem by a mathematical model, that is, a set of mathematical relationships (e.g., equalities, inequalities, logical conditions) which represent an abstraction of the real world problem. Usually, a mathematical model in optimization theory consists of four key objects:

- data [also called the constants of a model];
- variables (continuous, semi-continuous, binary integer) [also called decision variables or parameters];
- constraints (equalities, inequalities) [sometimes also called restrictions]; and
- objective function.

The data may represent cost or demands, fixed operation conditions of a reactor, capacities of plants and so on. The variables represent the degrees of freedom, i.e., what it is necessary to decide: How much of a certain product is to be produced, whether a depot is closed or not, or how much material will be stored in the inventory for later use. The constraints can be a wide range of mathematical relationships: algebraic, analytic, differential or integral. They may represent mass balances, quality relations, capacity limits, and so on. The objective function expresses the goal: minimize costs, maximize utilization rate, minimize waste and so on. Mathematical models for optimization usually lead to structured problems such as:

- linear programming (LP) problems,
- mixed integer linear programming (MILP) problems,
- nonlinear programming (NLP) problems, and
- mixed integer nonlinear programming (MINLP) problems.

Besides building a model and classifying the problem a solver is needed, i.e., a piece of software which has a set of algorithms implemented capable of solving the problem listed above.

What is discrete* optimization? Classical optimization theory (calculus, variational calculus, optimal control) treats those cases in which the variables represent continuous degrees of freedom, e.g., the temperature in a chemical reactor or the amount of a product to be produced. On the other hand, mixed integer, combinatorial or discrete optimization involves variables restricted to integer values, for example, counts (numbers of containers, ships), decisions (yes-no), or logical relations (if product A is produced then product B also needs to be produced).

What is the difference between simulation and mathematical optimization? In contrast to simulation (parameter values are fixed by the user, feasibility has to be checked separately, no prove on optimality), mathematical optimization methods search directly for an optimal solution and guarantee that the solution satisfies all constraints. While in optimization, feasible solutions are specified a priori and implicitly by the constraints, in simulation somebody has to ensure that only those combinations of parameter values are evaluated or considered which represent 'appropriate scenarios'.

What commercial potential is there in discrete optimization? To give some idea of the scope of mathematical optimization in helping organizations some recent examples are cited where benefits have been reported. Firstly, at Delta Airlines it is reported[52] that the use of an optimization model is expected to save the company $(US)300 million over a three year period. Secondly, a comprehensive mathematical programming model[5] used by the Digital Equipment Corporation (DEC), has helped DEC to save $(US)100 million. In a simple blending problem BASF-AG saved several hundred thousands of DM/year[35], (Section 5.2); in some production and distribution problems (Section 4.2 of this paper), even saving several millions of DM/year.

This paper is structured as follows: Section 2 provides an overview on areas in chemistry and related fields in which MIP has been used successfully or which are at least very promising. Section 3 is intended for a more technical interested audience and provides some background on the mathematical solution approaches used in discrete optimization. Here is also discussed briefly the aspects of parallel discrete optimization and stress the importance of good modelling practice being essential for solving discrete optimization problems. In Section 4, several applications in which the author has been involved are reviewed. The examples discussed illustrate the broad spectrum of possible applications and touches on some important points relevant to successful modelling. Finally some future areas which MIP might enter or new directions in which MIP might move are discussed, together with the implications which can be expected by using discrete optimization focusing on the perspectives of that field.

2 A SURVEY OF REAL WORLD PROBLEMS

This section considers some areas in which applications of (linear and nonlinear) mixed integer optimization are found. While the list is certainly not complete, the real world problems mentioned are typical for the process industries although many additional applications also occur in other businesses and industries:

- production planning
(production, logistics, marketing)—MILP, MINLP;
- sequencing problems
(putting production into order)—MILP;
- scheduling problems
(production of goods requiring machines and/or other resources);
- allocation problems
(e.g., allocating resources to orders, people to tasks);
- distribution and logistics problems
(supply chain optimization)—MILP;
- blending problems
(production and logistics)—LP, MILP, NLP, MINLP;
- refinery planning and scheduling
(refineries, chemical process industry)—NLP, MINLP;
- process design
(chemical process industry, food industry, refineries)—MINLP;
- engineering design
(all areas of engineering)—NLP, MINLP;
- selection and warehouse/depot location problems
(strategic planning)—MILP;
- investment and de-investment design problem
(strategic planning)—MILP;
- network design
(planning, strategic planning)—MILP, MINLP;
- financial problems
(strategic planning)—MILP, MINLP.

While most of the problems indicated above can be solved with linear mixed-integer methods, problems occurring in the process industry very often lead to nonlinear mixed integer problems. Therefore, it is not surprising that strong efforts are made related to solving MINLP problems at research institutions in the chemical engineering community. The following list of references may at least cover some colleagues from the major centres of activity: C. Floudas at Princeton University[19], I. Grossmann at Carnegie Mellon University[16,25], N. Shah at Imperial College[51].

Among the problems there are a few classical ones: The fuel mixture problems in the refinery or petrochemical industry, include blending problems leading to the so-called pooling problem (see, for instance, Fieldhouse[17], or Chapter 11 in Kallrath and Wilson[35]), an almost classical problem in nonlinear optimization. It refers to the intrinsic nonlinear problem of forcing the same (unknown) fractional composition of multi-component streams emerging from a pool, e.g., a tank or a splitter in a mass flow network. Structurally,

* The terms mixed integer optimization or mixed integer programming (MIP) and discrete optimization are used synonymously in this article.

this problem is dominated by indefinite bilinear terms of the form $\Sigma_i \Sigma_j A_{ij} x_i y_j$ appearing in equality constraints. The pooling problem occurs in all multi-component network flow problems in which the conservation of both mass flow and composition is required. In the chemical process industry, reaction kinetics might lead to analytic nonlinearities, such as exponential nonlinearities due to Arrhenius terms of the form $e^{-\Delta \varepsilon / kT}$ describing the reaction kinetics. If, in addition, it has to be considered that plants operate in discrete modes, or that connections between tanks and cracker or tanks and vacuum columns have to be chosen selectively, then mixed-integer nonlinear optimization problems need to be solved. Process network flow or process synthesis problems[25] usually fall into this category, too. Examples are heat exchanger networks, distillation sequencing or mass exchange networks.

3 MATHEMATICAL BACKGROUND ON MIXED-INTEGER OPTIMIZATION

This section provides some of the mathematical and algorithmic background on mixed integer optimization. It is addressed to a more technical audience, and might be skipped by some readers who, however, should at least keep in mind that discrete optimization involves great knowledge on efficient modelling and that it helps companies to keep such know-how in-house. Those who are active in problem solving need to know a lot about good modelling practice and how to connect models and algorithms. For that reason, it is strongly recommended that practitioners and consultants develop a deep understanding of algorithms.

Besides the exact algorithms in Section 3.1, there also exist heuristic methods which can find feasible points of optimization problems but can only prove optimality or evaluate the quality of these feasible points when used in combination with exact approaches. Such methods include Simulated Annealing, Tabu Search, Genetic Algorithms, Evolution Strategy, Ant Colony Optimization and Neural Networks. However, since these methods are not optimization methods in the strict sense, there will be no further focus on them.

3.1 Definition of Mixed Integer Nonlinear Programming (MINLP) Problems

For vectors $x^T = (x_1, \ldots, x_{n_c})$ and $y^T = (y_1, \ldots, y_{n_d})$ of n_c continuous and n_d discrete variables, the augmented vector, $x_\oplus^T = x^T \oplus y^T$, an objective function $f(x, y)$, n_e equality constraints $h(x, y)$ and n_i inequalities constraints $g(x, y)$, an optimization problem:

$$\min \left\{ f(x,y) \;\middle|\; \begin{array}{l} h(x,y) = 0, \;\; h : X \times U \to \mathbb{R}^{n_e}, \; x \in X \subseteq \mathbb{R}^{n_c} \\ g(x,y) \geq 0, \;\; g : X \times U \to \mathbb{Z}^{n_i}, \; y \in U \subseteq \mathbb{Z}^{n_d} \end{array} \right\} \tag{1}$$

is called mixed integer nonlinear programming (MINLP) problem, if at least one of the functions $f(x, y)$, $g(x, y)$ or $h(x, y)$ is nonlinear. The vector inequality, $g(x) \geq 0$, is to be read component-wise. Any vector $x_\oplus^T$ satisfying the constraints of (1) is called a feasible point of (1). Any feasible point, whose objective function value is less or equal than that of all other feasible points is called optimal solution. From this definition it follows that the problem might not have a unique optimal solution.

The continuous variables in (1) could, for instance describe the states (temperature, pressure, etc.), flow rates or design parameters of plant or chemical reactors. The discrete variables, often binary variables, may be used to describe the topology of a process network or to represent the existence or non-existence of plants. Considering the following pure integer nonlinear problem with two integer variables y_1 and y_2:

$$\min_{y_1, y_2} \left\{ 3y_1 + 2y_2^2 \;\middle|\; \begin{array}{l} y_1^4 - y_2 - 15 = 0 \\ y_1 + y_2 - 3 \geq 0 \end{array}, \quad y_1, y_2 \in U = \mathbb{N}_0 = \{0, 1, 2, 3, \ldots\} \right\}$$

A feasible point is $y_1 = 3$ and $y_2 = 66$. The optimal solution $y^* = (y_1, y_2)^* = (2, 1)$ and $f(y^*) = 8$ is unique. Depending on the functions $f(x, y)$, $g(x, y)$ and $h(x, y)$ the following structured problems are obtained.

With a matrix A of m rows and n columns, i.e., $\mathsf{A} \in \mathcal{M}(m \times n, \mathbb{R})$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ and $n = n_c + n_d$. Since some problems occur as subproblems of others, it is very important that the algorithms to solve the subproblems are well understood and exploited efficiently. While LP problems can be solved relatively easily (the effort to solve an LP problem with m constraints requires approximately m iterations), the computational complexity of MILP and MINLP grows exponentially with the n_d. Numerical methods to solve NLP problems work iteratively and the computational problems are related to questions of convergence, getting stuck in 'bad' local optima and availability of good initial solutions. Global optimization applies to both NLP and MINLP problems and its complexity again increases exponentially in the number of variables.

3.2 Linear Programming

The feasible region of an LP problem is a polyhedron S. The optimal solution of an LP problem always lies on a vertex, i.e., on an extreme point of S. This fact is exploited by one of the best known algorithms for solving LPs, the simplex algorithm of G. B. Dantzig[14,15] which can be understood as a vertex-following method, i.e., as an algorithm which moves from one corner (vertex) of S to a 'new' corner by advancing along one edge at a time. The algebraic platform is the concept of the basis $\mathcal{B}$ of A, i.e., a collection $\mathcal{B} = \{A_{j_1}, \ldots, A_{j_m}\}$ of linearly independent columns of A; basic (or dependent) variables, x_b, are computed from the inverse basis, i.e., by $x_b = \mathcal{B}^{-1} b$, while the nonbasic (or independent) variables are fixed at their bounds. The main ideas of the simplex algorithm are:

- pricing-out (identifying a nonbasic variable as a candidate for a new basic variable),

Table 1.

acronym	type of optimization	$f(x,y)$	$h(x,y)$	$g(x,y)$	n_d	subproblem
LP	Linear Programming	$c^T x$	$\mathsf{A}x - b$	x	0	matrix inversion
MILP	Mixed Integer Linear Programming	$c^D x_\oplus$	$\mathsf{A}x_\oplus - b$	$x_\oplus$	≥1	LP
MINLP	Mixed Integer Nonlinear Programming				≥1	MILP, NLP
NLP	Nonlinear Programming				0	nonlinear equation solving
GLOBAL	Global Optimization				≥0	NLP, Branch & Bound

- eliminating a basic variable by the minimum-ratio rule, and
- linear algebra aspects (pivoting).

The model building process can positively influence the numerical performance of the subtasks. Here, a few key ideas are briefly mentioned to bear in mind:

- efficient pricing is supported by a strong driving force in the objective function,
- equations with many zero right-hand side entries can create difficulties when applying the MRR,
- sparse models are easier to solve than dense ones.

Most commercial solvers also provide pre-solvers (i.e., systems for analysing the model before attempting optimization). Such systems eliminate fixed variables, attempt to tighten bounds on variables, and try to perform other operations which enable the solvers to operate more efficiently.

Based on the results of Karmarkar, in the last few years a large variety of interior point methods (IPMs) has been developed[24,41]. The idea of IPMs is to move through the interior of the feasible region, i.e., to proceed from an initial interior point $x \in S$ satisfying $x > 0$, towards an optimal solution without touching the border of the feasible set S. Approaching the boundary of the feasible region is penalized, i.e., the condition $x > 0$ is guaranteed by subtracting a penalty term $\mu_k \Sigma_{i=1}^n \ln x_i$ from the original objective function $c^T x$. Thus, the original LP problem is solved by solving a sequence of logarithmic barrier problems:

$$\min_x c^T x - \mu_k \sum_{i=1}^n \ln x_i \quad \text{subject to} \quad \mathsf{A}x = b \quad \text{and} \quad \mu_k > 0.$$

By suitable reduction of μ_k, the weight of the penalty term $\mu_k \Sigma_{i=1}^n \ln x_i$ is successively reduced and the sequence of points, $x_*^{(k)}$, obtained by solving the perturbed problems, converges, for $k \to \infty$, to the optimal solution, x_*, of the original problem. However, IPMs will in general return an approximately optimal solution which is strictly in the interior of the feasible region. Unlike the simplex algorithm, IPMs do not provide an optimal basic solution. Thus, 'purification' pivoting procedures from an interior point to a vertex having an objective value which is no worse have been proposed, and cross-over schemes to switch from interior-point algorithm to the simplex method have been developed[3].

3.3 Mixed Integer Linear Optimization

Building mixed-integer models requires great caution. Often there exists different possibilities to formulate an optimization problem[8], sometimes adding redundant constraints makes an algorithm work faster. Even some nonlinear optimization problems can be transformed to MILPs using special types of discrete variables[20,35,55].

- Logical conditions, such as 'and', 'or', 'not', 'implies', and also disjunctive constraints are formulated with binary variables $\delta \in \{0, 1\}$;
- Binary variables can indicate the state of a continuous variable and at the same time impose upper and lower bounds (L and U) on this variable. The constraints $x = 0 \vee L \le x \le U$ defining a semi-continuous variable x are equivalent to $L\delta \le x \le U\delta$, where δ is a binary variable;
- Special Ordered Sets (SOS) have been developed to formulate common types of restrictions in mathematical programming. In SOS of type 1 variables at most one variable (continuous or integer) can be non-zero. In an SOS of type 2 set, at most two variables which are adjacent in the ordering of the set can have non-zero values. Such sets often are used to model piecewise linear functions, e.g., linear approximations of nonlinear functions;
- Expressions containing products of k binary variables can be transformed into MILP models according to:

$$\left\{\delta_p = \prod_{i=1}^k \delta_i\right\} \Leftrightarrow \left\{\delta_p \le \delta_i,\ i = 1, \ldots, k; \quad \sum_{i=1}^k \delta_i - \delta_p \le k - 1; \quad \delta_i \in \{0, 1\}\right\}. \tag{2}$$

- Problems with linear constraints and objective functions of the form $\Sigma_{j=1}^k f(x_j)$, in which the variables x_j can only assume values of a discrete set $X = \{X_1, \ldots, X_N\}$, can be replaced by MILP problems, if the nonlinear function f fulfills certain generalized 'convexity' conditions.

A great variety of algorithms to solve mixed integer optimization problems has arisen during the last few decades. Among the best known exact algorithms for solving MILPs are efficient implicit enumerative algorithms that include pruning criteria so that not all feasible solutions have to be tested for finding the optimal solution and for proving optimality. The widely used Branch & Bound (B&B) algorithm with LP-relaxation, first developed in 1960 by Land and Doig[38], is the most important representative of enumerative algorithms. The branch in B&B hints at the partitioning process—a 'divide-and-conquer' like approach—used to prove optimality of a solution. Lower* bounds are used during this process to avoid an exhaustive search in the solution space. The B&B idea or implicit enumeration characterizes a wide class of

* In a maximum problem, otherwise upper bounds.

algorithms which can be applied to discrete optimization problems in general.

The computational steps of the B&B algorithm are as follows: After some initialization, the LP relaxation that is that LP problem which results if all integer variables are relaxed to continuous ones—establishes the first node. The node selection is obvious in the first step (just take the LP relaxation), later on it is based on some heuristics. A B&B algorithm of Dakin[13] with LP relaxations uses three pruning criteria: infeasibility, optimality and value dominance relation. In a maximization problem, the integer solution found leads to an increasing sequence of lower bounds, while the LP problems in the tree decrease the upper bound. A pre-set addcut $\alpha \geq 0$ causes the algorithm to accept a new integer solution only if it is better by at least the value of α. If the pruning criteria fail branching starts: The branching in this algorithm is done by variable dichotomy: for a fractional y_j^* two son nodes are created with the additional constraint $y_j \leq [y_j^*]$ resp. $y_j \geq [y_j^*] + 1$. Other possibilities for dividing the search space are, for instance, generalized upper bound dichotomy or enumeration of all possible values, if the domain of a variable is finite[11,14]. The advantage of variable dichotomy is that only simple constraints on lower and upper bounds are added to the problem.

The search strategy plays an important role in B&B schemes, widely used is the depth-first plus back-tracking rule[44]. Another important point is the selection of the branching variable. A common method of choosing a branching variable is by user-specified priorities, because no robust general strategy is known. The B&B algorithm terminates after a finite number of steps, if the solution space of the MILP problem's LP-relaxation is bounded.

Alternatively, or in addition, cutting plane algorithms and Branch & Cut (B&C)[23,12,54,45,7,10] might be used to solve MILP problems. After computing the continuous optimum by LP-relaxation of the integrality constraints, step by step new linear valid inequalities (cuts) are added to the MILP. With the help of these cuts noninteger variables of the relaxed solutions are forced to take integer values[11,44]. Cutting plane methods are not restricted to MILPs, they are used, e.g., in nonlinear and nondifferentiable optimization as well (Lemaréchal in Nemhauser *et al.*[43]).

3.4 Nonlinear Optimization

For historical reasons, constrained nonlinear optimization problems are also referred to as nonlinear programming problems (NLP). It is assumed that the functions $f(x)$, $g(x)$ and $h(x)$ are continuously differentiable on the whole vector space $\mathbb{R}^n$. In nonlinear problems we have to distinguish between local optima and the global optimum. Loosely speaking, the global optimum is the best of all possible values while a local optimum is the best in a nearby neighbourhood only. In practice, it is observed that, for instance, the pooling problem has usually several local optima. Depending on the initial guesses the solver finds different local optima. Thus, solving models involving pooling problems requires great care and deep understanding of the underlying real world problems.

In the early 1950s Kuhn and Tucker[37] extended the theory of Lagrangian multipliers, used for solving equality constrained optimization problems, to include the NLP problem (formulated as a maximum problem in the original work) with both equality and inequality constraints. The key idea is to transform them into an unconstrained problem and to use the optimality condition defined above. Thus, the theory is based on the definition of a Lagrangian function:

$$\mathcal{L}(x, \lambda, \mu) := f(x) + \lambda^T h(x) - \mu^T g(x),$$

that links the objective function $f(x)$ to the constraints $g(x)$ and $h(x)$. The vectors variables $\lambda \in \mathbb{R}^m$ and $\mu \in \mathbb{R}^l$ are called Lagrange multipliers. They are additional unknowns of the problem. Necessary conditions for an local optimal solution to NLP problems are the (Karush-)Kuhn-Tucker-conditions.

If x^* is a local solution to an NLP problem, and the functions $f(x)$, $g(x)$, and $h(x)$ are differentiable, then there exists a set of vectors μ^* and λ^* such that x^*, μ^* and λ^* satisfy the relations:

$$h(x) = 0, \quad g(x) \geq 0, \quad \mu^T g(x) = 0, \quad \mu \geq 0,$$
$$\nabla \mathcal{L}(x, \lambda, \mu) = 0.$$

It should be noted that these conditions rely on a constraint qualification, i.e., a regularity assumption applied to the constraints, to hold true. While convex problem will satisfy the constraint qualification, for nonconvex problem this is not necessarily so.

Numerical algorithms to compute local optima of NLP problems basically reduce to the solution of sets of nonlinear equations. Therefore, they require initial guesses. These guesses should be provided with care to enable the algorithm to converge, and strongly depend on the problem. However, let us give at least one piece of general advice: setting initial guess variables to zero should be avoided. Once a good solution is found the values of the variables should be kept and should be used as initial values in further computations.

If f is convex, and if the feasible region S is a convex set, e.g., if h is linear and g is concave, this is a convex optimization problem. For this class of problems, local optimality implies global optimality[47], i.e., every local minimum of a convex NLP problem is a global one. Unfortunately, due to the presence of nonlinear equalities, most practical problem are not convex.

Most algorithms[22,18] used to compute local optima of NLP problems are based on Taylor series expansions terminated after the linear or quadratic term. Inequality conditions are included, for instance, by applying active sets methods. The most powerful nonlinear optimization algorithms are the Generalized Reduced Gradient algorithm (GRG), Sequential Quadratic Programming (SQP) methods and Interior Point methods (IPM)[9,56] for problems involving many inequalities. The GRG algorithm was first developed by Abadie and Carpenter[2], (more recent information is contained in Abadie[1], Lasdon *et al.*[40], and Lasdon and Waren[39], but see also Gill *et al.*[22], Section 6.3). A special class of IPM includes inequalities by adding logarithmic penalties terms to the objective function. Then the problem can be solved as a nonlinear optimization problem with equality but no inequality constraints. For NLP problems with only a few nonlinear terms and in particular NLP problems containing pooling problems, recursion or, more efficiently sequential linear programming (SLP), are frequently used. A typical case, in which recursion or SLP is applied are models involving product terms cx with flow rate x and concentration c. If c is known, the problem is linear, otherwise it is nonlinear. The key idea behind recursion is the following. Some coefficients, usually the concentrations c, in an LP problem are defined as functions of the optimal values of LP variables. When an LP problem has been solved, the coefficients are re-evaluated and the LP re-solved. Under some assumptions this process may converge to a local optimum. In SLP, the nonlinear product terms cx are replaced by their Taylor series approximations. Letting c_0 and x_0 be the current values of c and x, the first order approximation for cx is:

$$c \cdot x \cong c_0 \cdot x_0 + x_0 \cdot (c - c_0) + c_0 \cdot (x - x_0) = c_0 \cdot x + x_0 \cdot \Delta c,$$
$$\Delta c := c - c_0. \tag{3}$$

The right-hand side of (3) is linear and has the unknowns x and Δc. The pool concentration, c, acts as a nonlinear variable, whose change, Δc, is determined by solving the LP. This leads to a new value of c, determined by $c = c_0 + \Delta c$ used to initialize the next iteration, replacing c_0. Note that x acts as a special linear variable with nonconstant coefficient c_0. Most SLP implementations include bounds on all 'Δc' variables, of the form $-S \le \Delta c \le S$, where S is an appropriate step bound[57], imposed to ensure that the Taylor series approximations are sufficiently accurate and that the algorithm converges.

3.5 Parallel Mixed Integer Linear Optimization?

In order to solve complex MIP problems with not only a few hundred, but rather a few thousand, or even ten thousands of discrete variables, BASF initiated the project PAMIPS (Parallel Algorithm and software for Mixed Integer Programming in Industrial Scheduling) supported under the ESPRIT initiative of the European Community connecting four industrial partners and three universities[49].

The exact methods, briefly described in Sections 3.2 and 3.3 for solving MILP problems, provide two different ways for the parallelization: the combinatorial part of the algorithm and the LP algorithm. The tree structure in mixed integer optimization indicates that more benefits are to be expected from parallelizing the combinatorial part. The combinatorial part is either a B&B or a B&C algorithm. In both cases it is necessary to solve many LPs. Obviously, the evaluation of the subproblems may be performed by a network of parallel processors or workstations.

3.6 Mixed Integer Nonlinear Programming

MINLP problems such as (1) are the most difficult optimization problems of all. They belong to the class $\mathcal{NP}$-complete problems*. MINLP problems combine all difficulties of both its subclasses: MILP and NLP.

Even worse, in addition they have properties absent in NLP or MILP. While for convex NLP problems a local minimum is identical to the global minimum, it is found that this result does not hold for MINLP problems.

A simple deterministic method would be to list all combinations U of discrete variables y_i. Fix them, and solve the NLP generated by fixed y_i yielding a pair $(x_i, z_i = f(x_i, y_i))$. Then choose the pair with smallest z_i (referred to here using the index i^*). Thus, the solution is given by the triple $(x^* = x_i,\ y = y_{i^*}, z_i^* = f(x_{i^*}, y_{i^*}))$. This method, of course, works only efficiently if U has not too many elements and if the NLP subproblems allow us to determine their global minima. Efficient deterministic methods[19] for solving MINLP problems fall into three classes:

- Branch & Bound (B&B) by Gupta and Ravindran[26],
- Generalized Benders Decomposition (GBD) by Geoffrion[21], and
- Outer-Approximation (OA) by Duran and Grossmann[16].

The B&B algorithm for MINLP problems is based on similar ideas to the B&B algorithm for solving MILP problems. The first step is to solve the problem generated by relaxing the integrity condition on the variables. If the solution of that problem fulfills all integrality conditions, the whole problem is solved. Otherwise, in a minimization problem the relaxed problem provides a lower bound (of course only, if the global minimum can be determined) and a search tree is built up. A feasible integer solution provides an upper bound. A major drawback of the B&B algorithm applied to MINLP problems is that nodes deeper in the tree cannot benefit so greatly from information available at previous nodes, as is the case in MILP B&B algorithms using the dual Simplex algorithm.

GBD divides the variables into two sets: complicating and non-complicating variables. In MINLP models the class of complicating variables is made up by the discrete (usually binary) variables. Then the algorithm generates a sequence of NLP subproblems (produced by fixing the binary variables y^k) and solves the so-called MILP Master problems in the space of the complicating variables. The NLP subproblems yield upper bounds for the original problem while the MILP Master problems yield additional combination of binary variables y^k for subsequent NLP subproblems. Under convexity assumptions the Master problems generate a sequence of lower bounds increasing monotonically. The algorithm terminates when the lower and upper coincide.

OA also consists of a sequence of NLP subproblems (produced by fixing the binary variables y^k) generated by MILP Master problems. The significant difference is how the Master problems are defined. Algorithms based on OA describe the feasible region as the intersection of an infinite collection of sets with a simpler structure, e.g., polyhedra. In OA, the Master problems are generated by 'outer approximations' (linearizations, or Taylor series expansions) of the non-linear constraints in those points which are the optimal solutions of the NLP subproblems; that is a finite collection of sets. The key-idea of the algorithm is to solve the MINLP with a much smaller set of points, i.e., tangential planes. In convex MINLP problems, a superset of the feasible region is established. Thus, the OA Master problems (MILP problem in both discrete and continuous variables) produce a sequence of monotonically increasing lower bounds. The termination criterion is the same as above.

---

* That means that at present no algorithm is known which could solve (1) in polynomial time (polynomial in the problem size). If somebody would construct one, then this algorithm would also solve other problems in class $\mathcal{NP}$ in polynomial time[44]. Note that most MILP problems belong to the class of $\mathcal{NP}$-complete problems as well.

While the GBD Master problems have fewer variables and constraints, the OA algorithm provides tighter bounds and needs fewer iterations for convergence. Both GBD and OA are in many instances even capable of proving optimality and both have heuristic extensions for solving non-convex MINLP problems.

3.7 Global Optimization

The problem of the existence of multiple local optima in nonlinear optimization is treated in a mathematical discipline of its own: global optimization, a field including theory, methods and applications of optimization techniques aimed at detecting a global optimum of nonlinear problems. Global optimization applies to both, NLP and MINLP. Problems analysed in the context of global optimization are seen to be attacked by stochastic and deterministic solution approaches. In the group of stochastic methods there are genetic algorithms, evolution strategies, tabu search and simulated annealing. In the strict sense these methods are not optimization methods at all because they guarantee neither optimality nor a certain quality of feasible points they may find. In contrast, deterministic methods are based on progressive and rigorous reduction of the solution space until the global solution has been determined with a pre-given accuracy. Deterministic methods may be classified as primal-dual methods, interval methods, and B&B methods; they have in common that they derive upper and lower bounds including the objective function value of the global optimum. These methods are quite different from the classical concepts of gradients and Hessians. Typical B&B methods, for instance, exploit convex underestimators of the objective function and convexify the feasible region; they divide the feasible region into subregions, solve the NLP problem on that reduced set, derive bounds and branch, if necessary. While the B&B method in MILP problem is known to terminate after a finite number of steps, B&B methods in global optimization are subject to convergence proofs. However, if we want to find the global optimization only up to a small number $\varepsilon > 0$ between the upper and lower bound, then the B&B will also terminate after a finite number of steps. Not only B&B methods, but all deterministic algorithms in global optimization have more in common with what is used in discrete optimization rather than in nonlinear continuous optimization. So far, the problems which can be solved at present are specially structured and are usually small involving only up to, say a hundred or a thousand of variables or constraints but the field is growing and it is worthwhile to get into contact with; it may knock at your door tomorrow anyway. The *Journal of Global Optimization* or the books Horst and Tuy[30], Horst *et al.*[29] or Horst and Pardalos[28] are good and recommended starting points.

3.8 Efficient Problem Solving and Good Modelling Practice

The solution time of MIP or nonlinear problems can often be reduced significantly by appropriate modelling. This is important since in contrast to ordinary LPs, effective solution of MIP or nonlinear problems depend critically upon good model formulation, the use of high level branching constructs, control of the B&B strategy, scaling and the availability of good initial values. The solution times can be greatly influenced by observing a few rules which distinguish 'bad' from 'good' modelling. Good formulations in MIP models are those whose LP relaxation is as close as possible to the MILP relaxation, or, to be precise, those whose LP relaxation has a feasible region which is close to the convex hull, that is the smallest polyhedron including all feasible MILP points. In practice, this means, for example, that upper bounds should be as small as possible. If α_1 and α_2 denote integer variables, the inequality $\alpha_1 + \alpha_2 \leq 3.7$ can be bound-tightened to $\alpha_1 + \alpha_2 \leq 3$. Another example, the formulation (2) is superior to the alternative set of only two inequalities:

$$\left\{ \delta_p = \prod_{i=1}^{k} \delta_i \right\} \Leftrightarrow \left\{ \delta_p \leq \frac{1}{k}\sum_{i=1}^{k} \delta_i;\ \frac{1}{k}\sum_{i=1}^{k} \delta_i - k - \sum_{i=1}^{k} \delta_i \Bigg) \leq \delta_p \right\} \quad (4)$$

because the $k + 2$ inequalities in (2) couple δ_p and the individual variables δ_i directly.

In many models in addition, it might be possible, to derive special cuts, i.e., valid inequalities cutting off parts of the LP relaxation's feasible set but leaving the convex hull unchanged. If inequalities are detected of the form $x + A\alpha \geq B$ with constants A and B, and variables $x \in \mathbb{R}$ and $\alpha \in \mathbb{N}$ in the model, the model can be enhanced and tightened by the cuts:

$$x \geq [B - (C - 1)A](C - \alpha), \quad C := \left\lceil \frac{B}{A} \right\rceil = ceil\left(\frac{B}{A}\right), \quad (5)$$

where C denotes the rounded-up value of the ratio B/A. If a semi-continuous variable σ, e.g., $\sigma = 0$ or $\sigma \geq 1$, occurs in the inequality $x + A\sigma \geq B$ it can be shown that $x/B + \sigma \geq 1$ is a valid inequality tightening the formulation.

Preprocessing can also improve the model formulation. Preprocessing methods introduce model changes to speed up the algorithms. They apply to both pure LP problems and MILP problems but they are much more important for MILP problems. Some common preprocessing methods are: presolve (logical tests on constraints and variables, bound tightening), disaggregation of constraints, coefficient reduction, and clique and cover detection[35].

Good modelling practice[55,35] takes advantages from good use of Presolve, Scaling and Branching Choice. Modelling appears more as an art rather than science. Experience clearly dominates. Remodelling and reformulations of problems (see, for instance, Kallrath and Wilson[35] Section 10.4) can significantly reduce the running time and the integrality gap (6), i.e., in a maximization problem the difference between the LP-relaxation and the best integer feasible point found. This task is still largely the responsibility of the modeller, although work has been done on automatically reformulating mixed zero-one problems[54]. Especially in nonlinear problems it is essential that the mathematical structure of the problem is exploited fully. It is important that good initial values are made available by, for instance, exploiting homotopy techniques.

4 BRIEF DISCUSSION OF SOLVED REAL WORLD PROBLEMS

This section briefly presents three solved real world problems in which the author has been involved; many more are found in Kallrath and Wilson[35] and Timpe and Kallrath[53] but even this small collection might give a little taste of what problems can be tackled successfully by mixed integer optimization. The first problem is only of small size but it has a complicated structure which requires the use of special cuts to obtain good solutions. The second one, similar to the one described in Timpe and Kallrath[53], is a very large problem but can be solved within minutes if its structure is exploited and some cuts developed. The third one is a process engineering problem which is partly a production planning problem being used operatively, but to some extent it is also a process design problem. A fourth problem, a site analysis, is briefly mentioned in Section 6.4.

It has to be said that in many practical cases optimality is not proven. Instead, the objective function value x_* of the optimal solution is bounded by $z^{LB} \leq z_* \leq z^{UB}$, where in a maximization problem z^{UB} follows from the LP relaxation, and z^{LB} is given by the objective function value z^{IP} of the best integer feasible point found. In the worst case, i.e., if $z_* = z^{UB}$, the relative error of z^{IP} with respect to z_* is given by the integrality gap Δ. In an maximization problem Δ is defined as:

$$\Delta := 100\frac{z^{UB} - z^{LB}}{z^{IP}}, \qquad z^{LB} = z^{IP}. \tag{6}$$

In most practical cases and if a good model formulation or special cuts are added to the model, Δ is less than a percent.

4.1 A Blending Problem Including Discrete Features—A Small Size MILP Problem

In the chemical industries, raw materials with random ingredient concentrations have to be blended to get homogeneous (intermediate) products. The characteristics and properties of blended products depend on the portions of raw materials and their concentrations of ingredients. Often linear blending behaviour can be assumed.

This blending model contains special features of process scheduling; for the full mathematical model see Reuter[48] and Kallrath[31]. The blended products are not bulk products but individual, ordered blends satisfying customer specified upper and lower concentrations of ingredients. The charges of raw material and the blends are handled in containers of equal size. The blending process occurs in a filling plant that allows the drawing of material from one container into another. To produce one or more containers of a blend, first the appropriate raw material containers have to be selected, and the masses to be taken from them have to be determined. Filling strategies and drawing losses are also considered in the model.

The problem is formulated as a MILP model. The objective is to minimize the costs of raw material, labour of the process and drawing losses. The constraints are divided into the following blocks:

- Blending constraints: lower and upper bounds on concentrations, balance equations for materials,
- Scheduling constraints: assignment of the raw material containers to blending orders and transportation,
- Logical cuts (special valid inequalities derived from the rules of container handling).

Using a straightforward model formulation, the problem could not be solved within an hour on a 486/33 IBM PC. Even for small problems optimality could not be proven because of the combinatorial structure of container choice, filling out mass determination and container scheduling. However, adding logical cuts to the model, even for larger problems, e.g., the production of 10 different blends of customer orders from an inventory of 60 raw material charges, the integrality gap was reduced to less than 2.5%. The model consists of 170 continuous, 660 semi-continuous and 1300 binary variables, 10 special ordered sets of type 1 and 2300 constraints. The first feasible integer solution is usually found after 10 minutes. The problem is solved with `XPRESS-MP`[6], data input and output, solution presentation and scheduling of the solver is managed by `MS-EXCEL`. This planning tool has been extensively used over the last four years.

4.2 Supply Chain Management—A Large MILP Problem

At BASF, a MILP model has been developed which led to a multi-site, multi-product, multi-period Production/Distribution Network Planning System, aiming to determine a production schedule for three months on a monthly basis in order to meet the given demand. The system is characterized by 7 production sites embracing 27 production units operating in fixed-batch-size mode, and subject to minimal production requirements[33]. 450 products were categorized as purchasable (225), producable (270), saleable (230), and consumable (60) and three time periods (usually, 3 months). Saleable products are sold to customers through 37 sales areas. These areas represent geographical locations of customer groups. Some customers attach the attribute to their demands that their product has to be produced at a specific production site. Therefore, to satisfy such demands, the origin of products must be known. Products produced on individual units remain 'unwrapped' (or bulk), others may be wrapped in two different ways. Purchased products may be available in any wrapping type. It is possible to rewrap at production sites products from any type of wrapping to any other type. Rewrapping is limited only by a rewrapping capacity of the site and available wrapping resources.

Products can be sorted at inventories located at the production sites. At present there is no inventory available at the sales areas but this is planned for the future. In order to guarantee its availability and being able to sell a product in a given period, a prespecified percentage of the product demand must be stored in an inventory by the end of the previous time period. This is modelled by considering a lower bound on the inventory for each product specified by the demand of the product in the following time period. Transport is possible between some of the production sites for some intermediate products, and from production sites to the sales areas for the finished products. There is no transport between sales areas.

Production occurs according to given recipes for each product. Typically, there are one or two recipes for each product-unit combination. The production is subject to resource and raw material availability. There is also a second quality product that can be sold. Finally, there are preferred production units for products. Products are manufactured in fixed product and unit dependent batch sizes. To produce a batch takes 0.25–1 day. It is not necessary to keep the batch requirement for the second and third period since the model is rerun every month. In addition, each production run must produce at least a minimum quantity of the product. Such minimal production runs take 0.5–10 days.

The model contains blocks of constraints representing inventory balances, sales, purchases, production and wrapping, inventory, transport, and the objective function.

The problem has 100,000 constraints, 150,000 variables and 2500 integers. The first integer feasible solution, using again XPRESS-MP, is usually found[6] within a few minutes on a Pentium II. Applying some special preprocessing techniques operating directly on the input data stored in the database* reduces the problem size to 70,000 constraints, 80,000 variables, 112 integers, 652 semi-continuous variables and a total of 230,000 matrix elements. The first integer feasible solution is now usually found within a minute. Using the special cuts (5) derived from this model the integrality gap Δ falls below 1%, and in some cases it is even possible to prove optimality in less than 15 minutes.

The advantage the client sees in this production planning system is, besides financial savings, to be able to handle the complexity of the problem and to do sensible planning at all. There is one very unique feature about the use of the production planning system. This feature is strongly recommended for other projects since it increases the acceptance of the approach significantly: production planning is done on-line with a responsible representative of each production site and one team member of BASF's mathematical consulting group joining the monthly meeting. The production plan is accepted after discussions which may add further constraints to the model. The place of the meeting changes so that each place and production site is treated equally. Thus, there is no central force dominating the production planning process.

4.3 Nonlinear Mixed Integer Problems

A variety of MINLP models in the chemical process industry have been discussed by Kallrath[32]: a production planning problem in BASF's petrochemical division, a tanker refinery scheduling problem, a site analysis of one of BASF's bigger sites, and a process design problem in which some process parameters and the optimal connections between up to 30 reactors are computed with respect to optimal total production, selectivity, energy, and costs. Nonlinearities are related to the exponential terms for the reaction kinetics and mass fractions used to interpolate density and viscosity of the fluid inside the reactors. Discrete variables are required to model minimum flow rates between reactors, the number of reactors and their connections; continuous variables represent flow rates. The optimization model has been embedded into an attractive and easy to use user-interface. It helps the client in daily production planning duties to adjust the plant immediately to current needs, i.e., changes in costs, capacity fluctuations or to attributes of orders. The tool supports the process design phase and helps to design cascades and connections of a system of reactors. The new designs save raw material, minimize waste material and increase the capacity of the reactor system.

* This involves tracking and resolving the supply chain within the database, and eliminating certain superfluous or inconsistent data and removing them from the data stream to the matrix generator. For example, data for a forced production of products for which no demand exist, are not passed. The advantage is that the time required to transfer the data to the matrix generator, the time to generate the matrix, and the time the solver spends in pre-solving is reduced.

5 MATHEMATICAL OPTIMIZATION IN PRACTICE

This section summarizes some of the experiences gathered by the author over the last 10 years. Readers more interested in the Section 'Good Modelling Practice' are further referred to Williams[55] and Kallrath and Wilson[35].

5.1 Benefits to Users of Mathematical Optimization

Users of mathematical optimization benefit in three major ways when models are developed and problems are solved. Firstly, there is the gain through the greater understanding of the problem. The very act of working through a model formulation with its builder can be of considerable benefit to a client. Secondly, there is the decision support system to be developed using the model and its solution capability. Using such a system can lead to substantial savings. Thirdly, there is the availability of a model for future experimental purposes. It is possible to test ideas for future planning on the model that could not be conducted on the actual processes themselves. Thus, the gains to clients are considerable. The prospect of saving even a few percent of a very large cost is exciting for a client. Similarly, the prospect of devising a new business process which will make an organization more competitive can be very encouraging and justifies the mathematical optimization approach and related project costs.

5.2 Consistent and Obtainable Data

Mathematical programming models rely on data of consistent quality being readily available. This will not always be so. It may be glib to assume that information can be obtained on the cost of each of 20 stages of a production process. People asked for information may not even be able to agree on a definition of cost. Thus, the data are subject to uncertainties. This may lead to a lack of user confidence in the model.

What must be done is to devise the best procedures possible for data collection and to see that procedures are adhered to if the model is going to be used repeatedly on data which may change over time. New data must be collected under the same terms as the old, until a major overhaul of the model is undertaken. It must then be made clear to users of the model that results were obtained under certain assumptions and stress that results should still be helpful in providing decision support provided appropriate caution is maintained.

5.3 User Interface

Users of a model want to see the results expressed in the context of their world, with its associated jargon, styles and symbols. For them it will be far preferable to see 'increase cracker temperature to 800°C' as a solution value, rather than the bar $T_{17,21} = 800$. Thus, it will be incumbent on modellers to make good user interfaces with a high degree of acceptance. Of course, the modeller might be supported by a computer scientist who is an expert in programming user interfaces. However, in some cases it might be important to develop an individual user interface requested by the end-user; in other examples one might be prepared to hear that end-users want an interface link to systems such as SAP.

5.4 Communication

Even with a seemingly precise process such as mathematical optimization, there is a stage once modelling has been undertaken when the modeller has to 'sell the approach' to the client. This will be well before implementation takes place. The results of a modelling exercise may produce solutions which are perhaps unexpected, e.g., indicative of inefficient current practices and such results may prove unpopular with some individuals (but highly popular with others). Maybe the modeller did not understand the business issues correctly. Thus, the modeller has to convince the client that the model is performing according to the conditions laid down by the client and to establish that no stage has been omitted or information misinterpreted. Of course, the process of building a model is interactive requiring lots of communication. Therefore, it is important that a comfortable relationship of 'mutual trust' exists throughout the lifetime of a mathematical optimization project between modeller and client for whom the model is being built. The project described in Section 4.2 is a good example for a very constructive communication between client and modeller.

5.5 Communication and Validating Solutions

The process of building a model is interactive requiring lots of communication. The same is true for the modeller's task to convince the client that the model is performing according to the conditions laid down by the client, and to establish that no stage has been omitted or information misinterpreted. Emphasis switches from validation by the modeller to validation by the end-user. Once solutions are proposed, the validation process continues and the modeller must continue to work with the client. The model would normally be tried out on a test data pack to ensure that the model is robust, is predictively valid (produces predictions that are in line with existing possibilities) and is replicatively valid (produces working solutions). Thus, communication and feedback between client and modeller continues. When, finally, a proposed solution is considered for implementation, checking will continue as the process of implementation may not be straightforward. It will also be important for the modeller to stress what assumptions have been made in the modelling and the potential shelf-life of the model. The implementers and users will need to monitor the model in the future to see if any breaches of the assumptions are made or any aspects of the decision support system pass their expiry date.

5.6 Keeping a Model Alive

A number of difficulties exist in keeping a model alive. Firstly there are the typical difficulties with the use of any sophisticated software system. The user of the model may not follow the 'rules' envisioned by its developer and may try to use data that are not appropriate or to use the model for purposes for which it was not designed. Secondly, one has to retain the understanding of the model, its underlying assumptions and limitations at any time. Otherwise, the probability of getting the wrong results arises. This also indicates that using modelling and optimization requires substantial know-how on both the modeller's and user's side. Thirdly, there will be a danger that as time moves on new users will emerge who will feel that as they were not involved in the original development of the model, they can have no faith in it. This may have to be remedied by the involvement of the developers of the model from time to time.

The above three points suggest that even if a company decides to have a model developed by external consultants, it must plan to acquire or buy in the expertise for later validation of the model, even if it is continuing to produce apparently sensible results year after year. Communication with and feedback from the client is one of the most important necessary conditions for successful modelling and this leads to a valid model being built and used over years.

6 FUTURE PATHS OF MIXED INTEGER OPTIMIZATION

This section focuses on some likely paths and directions in which mixed integer optimization might move.

6.1 Solving Scheduling Problems Effectively

There is one class of optimization problems, namely scheduling problems, whose complexity can easily exceed today's hardware and algorithmic capabilities. Nevertheless, there are numerous promising contributions[51,50,36,50,46,4], especially from researchers in the chemical engineering community. What makes scheduling problems so difficult? Using exact methods such as MIP, in some cases it is not even possible to find feasible integer solutions because feasible integer solutions exist often only very deep in the tree. In many cases it is very difficult to derive useful upper and/or lower bounds despite the fact that the best formulations had been used. The problem with scheduling problems is that they usually lead to poor LP relaxations. Resource constraints can easily be fulfilled with fractional value of the binary variable used in time-indexed formulations, and thus lower bounds are very weak. Even using the parallel algorithms and powerful hardware, scheduling problems might be too complex, and cannot be solved with exact methods, at least not yet. In such cases, the last resort might be to use heuristic approaches, e.g., simulated annealing, tabu search, or as in the case discussed below, constraint programming (CP). Heipcke[27] successfully investigated a very difficult scheduling problem (Kallrath and Wilson[35], Section 10.5), which became a benchmark problem in both the MIP and CP community is described in the Section; it applied both methods, CP versus MILP. CP[42] was developed in the 1980s out of Logic Programming and Constraint Solving and was applied successfully to a large range of industrial applications, especially to discrete (optimization) problems such as planning and scheduling. CP is a technique for discrete optimization that uses a tree search and performs domain reduction at each node. CP models typically consist of a wide range of constraint types, e.g., special global constraint operators such as all different or cumulative allow a CP model to be expressed in a more compact form than for MIP problems.

Could MIP models be helpful in scheduling refineries? Refinery scheduling leads to MINLP problems due to the presence of the pooling problem. The scheduling problem usually comes up after a production planning problem has been solved. The data generated by the production planning problem are input data to the scheduling problem. The purpose of the scheduling problem is to transform the production plan into a schedule useful for all operations within a time horizon of a few days. In that sense the scheduling problem is more a feasibility problem than an optimization problem. While in typical production scheduling problems degeneracy and symmetry cause great problems, the nonlinear features in the refinery scheduling problems could destroy some symmetry and may lead to useful relaxations.

6.2 Hybrid Approaches: The Happy Marriage of MIP and CP

A trend over the last few years indicates that the mathematical and the constraint programming community approach each other. This may result in a hybrid approach, i.e., in language and algorithms combining elements from both communities. This may have a great impact on supply chain problems and scheduling. Just recently, the European Commission awarded the project LISCOS (Large Integrated Supply Chain Optimization Software) with several million Euros. The technical core of this project initiated by BASFs mathematical consultant group and 10 other partners is the development of MIP-CP hybrid techniques.

6.3 Solving Design and Operative Planning Problems Simultaneously

It is a frequent experience that clients ask for support on a production planning or scheduling problem for a plant or reactor which just went on-line in production. Often, especially in scheduling problems, it turns out that there exist certain bottlenecks. It would greatly improve the situation if the design of a plant or reactor was analysed simultaneously with the planning or scheduling problem. Certainly, this problem is mathematically demanding because scheduling problems alone are already very difficult to solve; one reason why scheduling problems are often difficult to solve lies in the fact of limited resources (raw material, machine availability, or personnel). However, if the design and planning/scheduling problem are part of one embracing model, the bottleneck situation might be avoided. This simultaneous approach requires the availability of a realistic and detailed forecast of demand data and that the departments responsible for the design and the planning/scheduling cooperate, the latter problem is by far the more difficult one, especially in large companies.

6.4 Solving Strategic and Operative Problems Simultaneously

Imagine a company running a production network like the one described in Section 4.2. The company wishes to buy additional plants, wants to open some new reactors based on improved technology, or to shut down some older reactors. In chemical multi-stage production there might exist logical connections between the status of certain reactors. The data governing such decisions are the cost to buy a plant, or to open or shut down a reactor. The investment or deinvestments should be sound over a time horizon of, say, up to 15 years. The best approach to analyse such situations is to develop a model like the one sketched in Section 4.2, and to enhance it by additional design plant or design reactors (leading to design variables and constraints) and let the model provide suggestions on optimal design decisions. Regarding the database, it is necessary to provide the full data set (recipes, production rates and capacity, etc.) for all design plants or design reactors. All financial data should be discounted over the time horizon in order to support a net present value analysis. Some chemical companies actually take this combined strategic/design and operative planning approach[34] provided, that optimality is proven, it is an elegant approach, it saves huge amounts of money and also supports an analysis related to the stability of the solution or new design.

Another typical case of a large site design problem involving operative aspects (forecasted demands to be satisfied), fully described in Kallrath[32], is the optimization of a network of process units at a large production site connected by a system of pipes subject to design decisions, and the option to invest in new re-processing units. Especially if new sites are established in otherwise nonindustrial areas, e.g., some plants in South East Asia, appropriate models embracing strategic and operational designs almost certainly lead to great benefits. As described in the section above, the reason why this approach is taken only rarely, is not a mathematical or technical one, but rather has its reasoning in cultural and inter-departmental structures of large companies.

7 SOCIAL IMPACT, LIMITS AND FUTURE PERSPECTIVES OF MIXED INTEGER OPTIMIZATION

The real world problems successfully solved by large companies in the chemical, airline, refining and other industries demonstrate the huge potential for reducing costs, increasing efficiency and flexibility, and generally contributing to the effective management of the enterprise.

However, despite improved algorithms, hardware and software technology, the professional optimization specialist always faces the reality that once he has solved a client's problem the client will inevitably come back with a tougher problem. For instance, the client will increase the number of time periods in a multi-time period model or perhaps disaggregate some process that has been modelled fairly crudely into its separate components, thus rendering the problem bigger and generally harder to solve.

In practice, users seem to think about solution times in several possible bands. The first acceptable band is where the solution time for the problem is of the order of 12 hours, so one overnight run can be done per work day. There is, in practice, little benefit to be gained from reducing this to 8 hours since probably there will still only be one run possible per day. The next band of solution times is of the order of one hour where, if some time is allowed for the user to inspect the results and decide upon another scenario to analyse, it is probably possible to do two or three runs per day. The next band is where the solution times are of the order of a minute or so, at which point several benefits start to accrue. The first is that now the user can start to contemplate optimization almost 'on-line', i.e., to use it to adapt to the situation as data changes. The second benefit is that the user can rapidly analyse many scenarios and start to get a good understanding of how the solution changes as parameters change. The final band of solution times is where the optimization takes the order of a second. This is where on-line optimization really occurs.

How can the expectations of users, who have the need to solve ever larger problems in less time be accommodated? It has already been stressed that modelling is vital to practical optimization. It is just about possible to get away with poor modelling if the problem is a pure LP problem. As long as the model is correct, the implications of a poor (larger, redundant) model are likely just to be longer running times by a factor of perhaps 2 or 3. Not desirable, but not disastrous. But when moving to MILP or even MINLP problems such laxity cannot be afforded. Here the difference between a good and a poor formulation may not be 2 or 3, but perhaps 5 or 6 orders of magnitude increase in solution times. The problem becomes effectively insoluble.

It is the author's experience that analysis and hard work by experienced and expert modellers very often yields several orders of magnitude improvements in solution times for MILPs. Even when the problem is just too difficult to solve, the insights obtained by this analysis and hard work often give very good heuristics. One might anticipate that with growing hardware and software capabilities, the importance of the experienced modeller decreases. The opposite is true. Experienced modellers will become more important, because, when hardware and software capabilities grow there is a demand for more complex and realistic models because clients will ask for more details in the model. It is the modeller's task and responsibility to bring clients demands and mathematical programming reality to a fruitful liaison. In addition to better hardware, the modeller will be facing more intelligent algorithms implemented in commercial software, e.g., providing efficient Branch & Cut routines which requires that modellers really keep themselves up-to-date. Not only MILP but also MINLP in general will become tractable. Last, but not least, modellers will have more flexible modelling tools at their finger tips: Model generators supporting cutting plane algorithms, formulating optimization problems from graphically designed network flow problems, providing links to complete different solution algorithms, and allowing a switch between different solvers.

Despite the success observed when applying MIP, the support given to expert decision making by this technology is still far from being widely accepted. Very often, analysts experience great reservations when talking to people working in production, logistics or marketing. There is a psychological and/or cultural barrier. Experts are used to decision taking based on experience and heuristic rules which are difficult to express explicitly. The approach to achieve objective solutions which can be controlled on a quantitative basis is new. It may create unconscious fears, and may in addition require a huge effort to explain the problem of interest to a non-specialist with the appropriate degree of completeness and accuracy. Indeed, on the one hand the mathematical kernel of the application operates as a black box usually difficult to understand for non-mathematicians. On the other hand, experts are afraid to lose influence and acknowledgement when outsiders, in this case mathematicians, can produce solutions which prove to be better in terms of costs, contribution margin, utilization rate or some other valuable quantity, when compared to their solutions. Usually at least 50% of all efforts and time spent during project work trying to solve a real-world problem using mathematical optimization methods is related to psychology, i.e., talking to clients in order to increase the acceptance of the solution techniques or removing reservations and fears against mathematics.

8 SUPPORT AND CONSULTING FIRMS

Since mixed integer, and especially mixed integer nonlinear optimization problems are difficult to solve, one might expect that universities and research institutions engaged in optimization research and its applications, e.g., Abo Akademi, Carnegie Mellon, Imperial College, Princeton and Purdue are appropriate addresses to contact. Another, natural idea is to contact the software and tool developers such as

AMPL, Bell Labs (http://www.bell-labs.com/),
CPLEX, ILOG Consulting Group (http://www.ilog.fr/corporate/support/),
GAMS Inc., Washington D.C., US (http://www.gams.com),
LINDO Systems Inc. (http://www.lindo.com),
MathPro 2000 (http://sundown-vmp.com/mathpro),
OSL, IBM Business Consulting (http://www.ibm.com/services/buscon/),
XPRESS-MP, Dash Associates, Ltd., England, (http://www.dash.co.uk).

In addition, or alternatively, one might consult firms specializing on projects related to optimization.

ARKI Consulting and Development A/S, Denmark (e-mail to info@arki.dk),
MαBOS Mathematical Business Optimization Services GmbH, Germany (http://www.mabos.com),
Mathesis GmbH, Germany (http://www.mathesis.de),
NovaeTechnology Inc., US (http://www.novaetechnology.com),
TechnoLogix Decision Sciences Inc., Canada (http://www.technologix.ca).

These firms have clearly the advantage that they are not biased in terms of tools and software packages.

9 SUMMARY

An overview has been provided on real-world problems which can be successfully tackled by mixed integer optimization (MIP). Some of them have been discussed in detail and the importance of good modelling practice (e.g., ensuring that the linear programming relaxation is close to the convex hull, or that the initial values used in nonlinear problems lead to fast convergence) has been stressed. In particular, nonlinear problems are very demanding in terms of mathematical modelling, appropriate tuning of the algorithms (e.g., scaling), and the quality of the initial values computed, for instance, by special homotopy techniques.

It is shown that MIP can provide a quantitative basis for decisions and allow to be coped with most successfully. MIP has proven itself as a useful technique to reduce costs and to support other objectives. It is a technique under continuous development and has much to offer for the future. Some further areas MIP might enter or new directions in which MIP might move have been indicated: combined strategic and operative as well as combined design and operative modelling.

While MIP has already well established itself, further quantum leaps in practical optimization are to be expected. Global optimization of nonlinear problems knocks at our doors and might, say, within 5 to 10 years, play a similar role as does MIP nowadays.

REFERENCES

1. Abadie, J., 1978, The GRG method for nonlinear programming, in *Design and Implementation of Optimization Software*, H. J. Greenberg (editor), (Sijthoff and Noordhoff, The Netherlands) pp 335–363.
2. Abadie, J. and Carpenter, J., 1969, Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints, in *Optimization*, R. Fletcher (editor), (Academic Press, New York) pp 37–47.
3. Andersen, E. D. and Ye, Y., 1994, *Combining interior-point and pivoting algorithms for linear programming. Technical report*, (Department of Management Sciences, The University of Iowa, Ames, Iowa).
4. Applequist, G., Samikoglu, O., Pekny, J. and Reklaitis, G. V., 1997, Issues in the use, design and evolution of process scheduling and planning systems, *ISA Trans*, 36: 81–121.
5. Arntzen, B. C., Brown, G. C., Harrison, T. P. and Trafton, L. L., 1995, Global supply management at digital equipment corporation, *Interfaces*, 25: 69–93.
6. Ashford, R. W. and Daniel, R. C., 1995, *XPRESS-MP Reference Manual* (Dash Associates, Blisworth House, Northants NN7 3BX).
7. Balas, E., Ceria, S. and Cornuejols, G., 1993, A lift-and-project cutting plane algorithm for mixed 0-1 programs, *Mathematical Programming*, 58: 295–324.
8. Barnhart, C., Johnson, E. L. and Nemhauser, G. L., 1993, Formulating a Mixed-Integer Programming Problem to Improve Solvability, *Operations Research*, 41: 1013–1019.
9. Bazaraa, M., Sherali, H. D. and Shetty, C. M., 1993, *Nonlinear Programming*, 2nd edition, (Wiley, Chichester, UK).
10. Boyd, E. A., 1994, Fenchel cutting planes for integer programs, *Operations Research*, 42(1): 53.
11. Burkhard, R. E., 1972, *Methoden der Ganzzahligen Optimierung*, (Springer, Wien, New York).
12. Crowder, H. E., Johnson, E. L. and Padberg, M. W., 1983, Solving large scale 0-1 linear programming problems, *Operations Research*, 31: 803–834.
13. Dakin, R. J., 1965, A tree search algorithm for mixed integer programming problems, *Computer Journal*, 8: 250–255.
14. Dantzig, G. B., 1951, Application of the simplex method to a transportation problem, in *Activity Analysis of Production and Allocation*, T. C. Koopmann (editor), (Wiley, New York).
15. Dantzig, G. B., 1963, *Linear Programming and Extensions*, (Princeton University Press, Princeton, New Jersey).
16. Duran, M. A. and Grossmann, I. E., 1986, An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programms. *Mathematical Programming*, 36: 307–1986.
17. Fieldhouse, M., 1993, The pooling problem, in *Optimization in Industry: Mathematical Programming and Modeling Techniques in Practice*, T. Ciriani and R. C. Leachman (editors), (John Wiley and Sons, Chichester, UK).
18. Fletcher, R., 1987, *Practical Methods of Optimization*, 2nd edition (Wiley, Chichester, UK).
19. Floudas, C. A., 1995, *Nonlinear and Mixed Integer Optimization*, (Oxford University Press, Oxford, UK).
20. Garfinkel, R. S. and Nemhauser, G. L., 1972, *Integer Programming*, 2nd edition, (John Wiley, New York).
21. Geoffrion, A. M., 1972, Generalized Benders Decomposition, *J Optimization Theory and Applications*, 10: 237–260.
22. Gill, P. E., Murray, W. and Wright, M. H., 1981, *Practical Optimization*, (Academic Press, London).
23. Gomory, R. E., 1958, Outline of an algorithm for integer solutions to linear programming, *Bull Amer Math Soc*, 64: 275–278.
24. Gonzaga, C. L., 1992, Path-Following Methods for Linear Programming, *SIAM Review*, 34: 167–224.
25. Grossmann, I. E., Caballero, J. A. and Yeomans, H., 1999, Mathematical Programming Approaches for the Synthesis of Chemical Process Systems, *Korean J Chem Eng*, 68: 407–426.
26. Gupta, O. K. and Ravindran, V., 1985, Branch and Bound Experiments in Convex Nonlinear Integer Programming, Management Science, 31: 1533–1546.
27. Heipcke, S., 1995, *Resource Constrained Job-Shop-Scheduling with Constraint Nets*, Diplomarbeit, Katholische Universität Eichstätt, Mathem.-Geographische Fakultät, (Universität Eichstätt, Germany).
28. Horst, R. and Pardalos, P. M., editors, 1995, *Handbook of Global Optimization*, (Kluwer Academic Publishers, Dordrecht , Holland).
29. Horst, R., Pardalos, P. M. and Thoai, N. V., 1996, *Introduction to Global Optimization*, (Kluwer Academic Publishers, Dordrecht, Holland).
30. Horst, R. and Tuy, H., 1996, *Global Optimization: Deterministic Approaches*, 3rd edition (Springer, New York).
31. Kallrath, J., 1995, Diskrete Optimierung in der chemischen Industrie, in *Mathematik in der Praxis—Fallstudien aus Industrie, Wirtschaft, Naturwissenschaften und Medizin*, A. Bachem, M. Jünger and R. Schrader (editors), (Springer-Verlag, Berlin), pp 173–195.
32. Kallrath, J., 1999, Mixed-Integer Nonlinear Programming Applications, in *Operational Research in Industry*, E. L. Johnson, T. A. Ciriani, S. Gliozzi and R. Tadei (editors), (Macmillan, Houndsmill, Basingstoke), pp 42–76.
33. Kallrath, J., 1999, The Concept of Contiguity in Models Based on Time-Indexed Formulations, in *Scientific Computing in Chemical Engineering II*, F. Keil, W. Mackens, H. Voss and J. Werther (editors), (Springer, New York), pp 330–337.
34. Kallrath, J., 2000, Combined Strategic, Design and Operative Planning—Two Success Stories in Applying MILP and MINLP, *J Europ Operational Research*, accepted, in print.
35. Kallrath, J. and Wilson, J. M., 1997, *Business Optimisation Using Mathematical Programming*, (Macmillan, UK).
36. Kondili, E., Pantelides, C. C. and Sargent, R. W. H., 1993, A General Algorithm for Short-Term Scheduling of Batch Operations—I. Mixed Integer Linear Programming Formulation. *Compu Chem Eng*, 17: 211–227.
37. Kuhn, H. W. and Tucker, A. W., 1951, Nonlinear Programming, in *Proc Second Berkeley Symp Mathematical Statistics and Probability*, J. Neumann (editor), *Berkeley, CA, 1951, University of California*, pp 481–492.
38. Land, A. H. and Doig, A. G., 1960, An automatic method for solving discrete programming problems, *Econometrica*, 28: 497–520.
39. Lasdon, L. S. and Waren, A. D., 1978, Generalized Reduced Gradient Method for Linearly and Nonlinearly Constrained Programming, in *Design and Implementation of Optimization Software*, H. J. Greenberg (editor), (Sijthoff and Noordhoff, The Netherlands), pp. 363–397.
40. Lasdon, L. S., Waren, A. D., Jain, A. and Ratner, M., 1978, Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming, *ACM Trans Math Software*, 4: 34–50.
41. Lustig, I. J., Marsten, R. E. and Shanno, D. F., 1992, On implementing Mehrotra's Predictor-Corrector Interior-Point Method for Linear Programming, *SIAM Journal of Optimisation*, 2: 435–449.

42. Marriot, K. and Hallo, P. J., 1998, *Programming with Constraints—An Introduction*, MIT Press, Boston.
43. Nemhauser, G. L., Rinnooy Kan, A. H. G. and Todd, M. J., (editors), 1989, *Optimisation*, (North-Holland, Amsterdam).
44. Nemhauser, G. L. and Wolsey, L. A., 1988, *Integer and Combinatorial Optimization*, (John Wiley, New York).
45. Padberg, M. W. and Rinaldi, G., 1987, Optimization of a 532-city traveling salesman problem by branch and cut, *Operations Research Letters*, 6: 1–6.
46. Pantelides, C. C., 1994, Unified Frameworks for Optimal Process Planning and Scheduling, in *Proc 2nd Conf Foundations of Comp Aided Proc Op*, (CACHE Corp.), pp 253–274.
47. Papadimitriou, C. H. and Steiglitz, K., 1982, *Combinatorial Optimisation: Algorithms and Complexity,* (Prentice Hall and International, Englewood Cliffs, New Jersey).
48. Reuter, B., 1994, *Mischungsplanung einer pharmazeutischen Produktion*, (Studienarbeit, TH Darmstadt, Darmstadt, Germany).
49. Schreieck, A. and Lücking, F., 1995, *PAMIPS Esprit Projects 8755 and Marketing Management—Public Report*, Technical report, BASF-AG, ZX/ZC-C13, D-67056 Ludwigshaften, Germany.
50. Shah, N., Pantelides, C. C. and Sargent, R. W. H., 1993, Optimal Periodic Scheduling of Multipurpose Batch Plants, *Annals of Ops Res*, 42: 193–228.
51. Shah, N., 1999, Single- and Multisite Planning and Scheduling: Current Status and Future Challenges, *AIChE Symp Series, 320*, 94: 75–90.
52. Subramanian, R., Scheff, R. P., Jr, Quinlan, J. D, Wiper, D. S. and Marsten, R. E., 1994, Coldstart: Fleet assignment at Delta Air Lines, *Interfaces*, 24(1): 104–120.
53. Timpe, C. and Kallrath, J., 2000, Optimal Planning in Large Multi-Site Production Networks, *J of Europ Operational Research*, 126(2).
54. VanRoy, T. J. and Wolsey, L. A., 1987, Solving mixed integer programs by automatic reformulation, *Operations Research*, 35(1): 45–57.
55. Williams, H. P., 1993, *Model Building in Mathematical Programming*, 3rd edition, (John Wiley, Chichester).
56. Wright, S., 1996, *Primal-Dual Interior-Point Methods*, (Society for Industrial and Applied Mathematics, Philadelphia, PA).
57. Zhang, J., Kim, N. and Lasdon, L. S., 1985, An Improved Successive Linear Programming Algorithm, *Management Science*, 31: 1312–1331.

ACKNOWLEDGEMENTS

Thanks are directed to G. DeBeuckelaer, T. Maindl, K. Plitzko, P. Schmitt (BASF Corp., Mt. Olive, NJ, USA) and Horst Fichtner (Institut für Theoretische Physik IV: Weltraum- und Astrophysik, Ruhr-Universität Bochum, Germany) for their encouraging words and interest in this work. This article benefited greatly from their suggestions and comments. A first version of this article came out on a CD, and in case material appears similar, it is here reprinted with permission from *Chemical Engineering*.

ADDRESS

Correspondence concerning this paper should be addressed to Dr J. Kallrath, BASF-AG, ZDP/C, C13, D-67056 Ludwigshafen, Germany. (E-mail: josef.kallrath@basf-ag.de).

The manuscript was received 26 November 1999 and accepted for publication after revision 22 May 2000.