Polyhedral techniques in combinatorial optimization

Aardal, K.I.; van Hoesel, C.P.M.

Published: 01/01/1995

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Polyhedral Techniques in Combinatorial Optimization

K. Aardal, S. van Hoesel

RM/95/023
Polyhedral Techniques in Combinatorial Optimization

Karen Aardal
Department of Econometrics
Tilburg University
P.O. Box 90153
5000 LE Tilburg, The Netherlands
aardal@kub.nl

Stan van Hoesel
Department of Quantitative Economics
Limburg University
P.O. Box 616
6200 MD Maastricht, The Netherlands
s.vanhoesel@ke.rulimburg.nl

July 27, 1995

Abstract

Generally, combinatorial optimization problems are easy to formulate, but hard to solve. The most successful approaches, cutting plane algorithms and column generation, rely on the (mixed) integer linear programming formulation of a problem. The theory of polyhedra, i.e., polyhedral combinatorics, is the foundation of these techniques.

This manuscript intends to give an overview of polyhedral theory and its implications for cutting plane algorithms. We describe the major theoretical developments, like Gomory's general cutting plane algorithm, and the complexity of separation. Practical issues are illustrated on a diversity of well-known problems, like the traveling salesman problem and facility location, and on some generic problems like the knapsack problem, and vertex packing problem. The cutting plane approach extended with preprocessing techniques, and it is embedded in a branch-and-cut framework. Typical computational results are provided.
Combinatorial optimization deals with maximizing or minimizing a function subject to a set of constraints and subject to the restriction that some, or all, variables should be integers. Several problems that occur in management and planning situations can be formulated as combinatorial optimization problems, such as the lot sizing problem, where we need to decide on which time periods to produce, and how much to produce in these periods to satisfy customers demand at minimal total production, storage and setup costs. Another well-known combinatorial optimization problem is the traveling salesman problem where we want to determine in which order a "salesman" shall visit a number of "cities" such that all cities are visited exactly once and such that the length of the tour is minimal. This problem is one of the most studied combinatorial optimization problems, not because of its importance in the planning of salesmen tours, but because of its numerous other applications, both in its own right and as substructures of more complex models, and because it is notoriously difficult to solve. The combination of being easy to state, relatively easy to formulate as a mathematical programming problem, but computationally intractable is something a majority of combinatorial optimization problems have in common.

The computational intractability of most core combinatorial optimization problems has been theoretically indicated, i.e. it is possible to show that most of these problems belong to the class of NP-hard problems, see Karp (1972), and Garey and Johnson (1979). No algorithm with a worst-case running time bounded by a polynomial in the size of the input is known for any NP-hard problem, and it is strongly believed that no such algorithm exists. Therefore, to solve these problems we have to use an enumerative algorithm, such as dynamic programming or branch and bound, with a worst-case running time that is exponential in the size of the input. The computational hardness of most combinatorial optimization problems has inspired researchers to develop good formulations, and algorithms that are expected to reduce the size of the enumeration tree. To use information about the structure of the convex hull of feasible solutions, which is the basis for polyhedral techniques, has been one of the most successful approaches so far. The pioneering work in this direction was done by Dantzig, Fulkerson and Johnson (1954), who invented a method to solve the traveling salesman problem. They demonstrated the power of their technique on a 49-city instance, which was huge at that time.

The idea behind the Dantzig-Fulkerson-Johnson method is the following. Assume we want to solve the problem

$$\min \{cx \text{ subject to } x \in S\},$$

where $S$ is the set of feasible solutions, which in our case is the set of traveling salesman tours. Let $S = P \cap \mathbb{Z}^n$, where $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ and $Ax \leq b$ is a system of linear inequalities. Since $S$ is difficult to characterize, we could solve the problem

$$\min \{cx \text{ subject to } x \in P\}$$

instead. Problem (2) is easy to solve, but since it is a relaxation of (1) it may give us a solution $x^*$ that is not a tour. More precisely, the following two things can happen if we solve (2): either the optimal solution $x^*$ is a tour which means that $x^*$ is also optimal for (1), or $x^*$ is not a tour in which case it is not feasible for (1). If the solution $x^*$ is not feasible for (1) it lies outside the convex hull of $S$ which means we can cut off $x^*$ by identifying a hyperplane separating $x^*$ from the convex hull of $S$, i.e. a hyperplane that is satisfied by all tours, but violated by $x^*$. An
inequality that is satisfied by all feasible solutions is called a valid inequality. When Dantzig, Fulkerson and Johnson solved the relaxation (2) of their 49-city instance they indeed obtained a solution $x^*$ that was not a tour. By looking at the solution they identified a valid inequality that was violated by $x^*$, and added this inequality to the formulation. They solved the resulting linear programming problem and obtained again a solution that was not a tour. After repeating this process a few times a tour was obtained, and since only valid inequalities were added to the relaxation, they could conclude that the solution was optimal.

Even though many theoretical questions regarding the traveling salesman problem remained unsolved, the work of Dantzig, Fulkerson and Johnson was still a breakthrough as it provided a methodology that was actually not limited to solving traveling salesman problems, but could be applied to any combinatorial optimization problem. This new area of research on how to describe the convex hull of feasible solutions by linear inequalities was called polyhedral combinatorics. During the last decades polyhedral techniques have been used with considerable success to solve many previously unsolved instances of hard combinatorial optimization problems, and it is still the only method available for solving large instances of the traveling salesman problem. The purpose of this paper is to describe theoretical and computational aspects of polyhedral techniques and to partially survey the results that have been obtained by applying this approach.

A natural question that arises when studying the work by Dantzig, Fulkerson and Johnson is whether it is possible to develop a general scheme for identifying valid inequalities. This question was answered by Gomory (1958), (1960), (1963) who developed a cutting plane algorithm for general integer linear programming, and showed that the integer programming problem

$$\min \{ cx \mid x \in S \}$$

can be solved by solving a finite sequence of linear programs. Chvátal (1973) proved that all inequalities necessary to describe the convex hull of integer solutions can be obtained by taking linear combinations of the original and previously generated linear inequalities and then applying a certain rounding scheme, provided that the integer solutions are bounded. Schrijver (1980) proved the more general result that it is possible to generate the convex hull of integer solutions by applying a finite set of operations on the polyhedron describing the integer solutions, if this polyhedron is rational, but not necessarily bounded. The results by Gomory, Chvátal, and Schrijver are discussed in Section 1. Here we will also address the following two questions: When can we expect to have a concise description of the convex hull of feasible solutions? How difficult is it to identify a violated inequality? These questions are strongly related to the computational complexity of the considered problem, i.e. the hardness of a problem type will catch up with us at some point, but we shall also see that certain aspects of the answers make it possible to hope that a bad situation can be turned into a rather promising one.

When studying special problem classes, such as the traveling salesman problem, we want to develop specific families of valid inequalities that contain inequalities that can be proven to be necessary in the description of the convex hull of feasible solutions. Based on the various classes of valid inequalities we then need to develop separation algorithms, i.e. algorithms for identifying violated inequalities given the current solution $x^*$. In Section 2 we begin by describing families of valid inequalities for some basic combinatorial optimization problems, and the corresponding separation problems. These inequalities are important as they are often useful when solving more complex problems as well, either directly, or as a starting point for developing new, more general families of inequalities. Moreover, they represent different arguments that can be used when developing valid inequalities. We shall also give a partial survey of polyhedral results for combinatorial optimization problems.
Next to the theoretical work of developing good classes of valid inequalities and algorithms for identifying violated inequalities, there is a whole range of implementation issues that have to be considered in order to make polyhedral methods work well. One such issue is *preprocessing*. Important elements of preprocessing are to reduce the size of the initial formulation by deleting unnecessary variables and constraints, and to reduce the size of the constraint coefficients to make the instance numerically more attractive. In the course of strengthening the relaxation by adding valid inequalities we may also want to *delete* some of the previously added inequalities to avoid that the formulation grows too much. We may also want to work with a partial set of variables to speed up computations. Dantzig, Fulkerson and Johnson were able to find the optimal solution by adding valid inequalities only. In general we do end up in the situation where the current solution \( x^* \) is not feasible and where we are unable to identify an inequality violated by \( x^* \). We then have to start a branch and bound phase. For the branch and bound algorithm we must decide precisely how to create new subproblems, or nodes, in the search tree, as well as a suitable search strategy. It is also possible to add inequalities in every node of the tree, in which case we need to keep track of where in the tree the various inequalities are valid. All these issues are discussed in Section 3. To illustrate the computational possibilities of polyhedral techniques we present computational results for some selected problem types in Section 4.

Even though polyhedral combinatorics has been the foremost tool for computing large instances of a vast collection of combinatorial optimization problems it is not the only technique present, and depending on the problem type it may be preferable to choose a different method. We conclude our article by briefly mentioning alternative approaches to solving integer and combinatorial optimization problems.
1 Theoretical background

The integer linear programming problem (ILP) is defined as

$$\min \{cx : x \in S\}$$

where $$S = P \cap \mathbb{Z}^n$$ and $$P = \{x \in \mathbb{R}^n : Ax \leq b\}$$, We call $$P$$ the linear formulation of ILP. A polyhedron $$P$$ is rational if it can be determined by a rational system $$Ax \leq b$$ of linear inequalities. The convex hull of the set $$S$$ of feasible solutions, denoted $$\text{conv}(S)$$, is the smallest convex set containing $$S$$. A facet defining valid inequality is a valid inequality that is necessary to describe $$\text{conv}(S)$$, i.e. it is the "strongest possible" valid inequality. In Figure 1 we give an example of sets $$S$$, $$P$$ and $$\text{conv}(S)$$.

![Figure 1: P, S, and conv(S).](image)

If we know the linear description of $$\text{conv}(S)$$ we can solve the linear programming problem $$\min \{cx : x \in \text{conv}(S)\}$$ which is computationally easy. In this section we shall primarily address the issue of how difficult it is to obtain $$\text{conv}(S)$$. First we show that for rational polyhedra, and for not necessarily rational bounded polyhedra, we can generate $$\text{conv}(S)$$ algorithmically in a finite number of steps. In general however, there is no upper bound on the number of steps in terms of the dimension of $$S$$. We also demonstrate that it is very unlikely that $$\text{conv}(S)$$ of any NP-hard problem can be described by concise families of linear inequalities. Finally, we relate the complexity of the problem of finding a hyperplane separating a vector $$x^*$$ from $$\text{conv}(S)$$ or showing that $$x^* \in S$$ to the complexity of the optimization problem given $$S$$. In general these two problems are equally hard, but if we restrict the search of a separating hyperplane to a specific class, this problem might be polynomially solvable even if the underlying optimization problem is NP-hard.

1.1 Solving Integer Programming Problems by Linear Programming

What was needed to transform the procedure of Dantzig, Fulkerson and Johnson (1954) into an algorithm was a systematic procedure for generating valid inequalities that are violated by the current solution. Assume that we want to solve the variant of ILP where the integer vectors in $$S$$ are bounded and where all entries of the constraint matrix $$A$$ and the right-hand side
vector $b$ are integers. Gomory (1958), (1960) and (1963) developed a cutting plane algorithm based on the simplex method, for solving integer linear problems on this form. This was the first algorithm developed for integer linear programming that could be proved to terminate in a finite number of iterations. The basic idea of Gomory's algorithm is similar to the approach of Dantzig, Fulkerson and Johnson, i.e. instead of solving ILP directly we solve the linear programming (LP) relaxation $\min \{cx : x \in P\}$ by the simplex method. If the optimal solution to LP is integral, then we are done, and otherwise we need to identify a valid inequality cutting off $x^\ast$. Gomory developed a technique for automatically identifying a violated valid inequality and proved that after adding a finite number of inequalities, called Gomory cutting planes, the optimal solution is obtained. We shall illustrate Gomory’s technique by an example. Assume we have solved the linear relaxation of an instance of ILP, as described above, by the simplex method, and that one of the rows of the tableau reads

$$x_1 - \frac{1}{11}x_3 + \frac{2}{11}x_4 = \frac{36}{11}$$

where $x_1$ is a basic variable and variables $x_3$ and $x_4$ are non-basic, i.e. at the current solution $x_1 = 36/11$ and $x_3 = x_4 = 0$. We now split each coefficient in an integer and a fractional part by rounding down all coefficients. The integer terms are put in the left-hand side of the equation and the fractional terms are put in the right-hand side. Since all coefficients are rounded down, the fractional part of the variable coefficients in the right-hand side becomes nonpositive,

$$x_1 - x_3 - 3 = -\frac{10}{11}x_3 - \frac{2}{11}x_4 + \frac{3}{11}.$$  

In any feasible solution to ILP, the left-hand side should be integral. Moreover, all variables are nonnegative. Since the variables in the righthand side appear with nonpositive coefficients we can conclude that

$$\frac{3}{11} - \frac{10}{11}x_3 - \frac{2}{11}x_4 \leq 0,$$

and integer. (4)

We have argued that the inequality (4) is valid, i.e. it is not violated by any feasible integer solution. It is easy however to see that it does cut off the current fractional solution as $x_3 = x_4 = 0$. Let $\lceil x \rceil$ denote the integer part of $x$.

**Outline of Gomory’s cutting plane algorithm.**

1. Solve the linear relaxation of ILP with the simplex method. The current number of variables is $k$. If the optimal solution $x^\ast$ is integral, stop.

2. Choose a source row $i_0$ in the optimal tableau with a fractional basic variable. Row $i_0$ reads $\bar{a}_{i_0,1}x_1 + \bar{a}_{i_0,2}x_2 + \ldots + \bar{a}_{i_0,k}x_k = \bar{b}_{i_0}$. Let $a'_{ij} = a_{ij} - \lceil a_{ij} \rceil$, and $b'_i = b_i - \lceil b_i \rceil$. 

3. Add the equation $-a'_{i_0,1}x_1 - a'_{i_0,2}x_2 - \ldots - a'_{i_0,k}x_k + x_{k+1} = -b'_i$, where $x_{k+1}$ is a slack variable, to the current linear formulation, and reoptimize using the dual simplex method. If the optimal solution $x^\ast$ is integral, stop, otherwise $k \leftarrow k + 1$, go to 2.

In the outline above we have not specified how to choose the source row. To be able to prove that the algorithm terminates in a finite number of steps we have to make sure that certain technical conditions are satisfied. The technical details are omitted here but can be found in Gomory (1963) who gives two proofs of finiteness, and in Schrijver (1986), page 357.
Theorem 1 Gomory (1963). There exists an implementation of Gomory's cutting plane algorithm such that after a finite number of iterations either an optimal integer solution is found, or it is proved that $S = 0$.

A recent discussion on Gomory cutting planes can be found in Balas et al. (1994) who incorporate the cutting plane algorithm in a branch-and-bound procedure and report on computational experience.

Chvátal (1973) studied the more general version of ILP, where the integer vectors of $S$ are bounded and where the entries of $A$ and $b$ are real numbers. He showed that if one takes linear combinations of the linear inequalities defining $P$ and then applies rounding, and repeats the procedure a finite number of times, $\text{conv}(S)$ is obtained. After each iteration of the procedure we get a new linear formulation containing more inequalities. We again illustrate the procedure by an example. Let $G = (V, E)$ be an undirected graph where $V$ is the set of vertices and $E$ is the set of edges. Let $\delta(v) = \{e \in E : e \text{ is incident to } v\}$. A matching $M$ in a graph is a subset of edges such that each vertex is incident to at most one edge in $M$. The maximum cardinality matching problem can be formulated as the following integer linear programming problem.

$$\begin{align*}
\max & \sum_{e \in E} x_e \\
\text{s.t.} & \sum_{e \in \delta(v)} x_e \leq 1 \quad \text{for all } v \in V \quad (5) \\
& 0 \leq x_e \leq 1 \quad \text{for all } e \in E \quad (6) \\
& x_e \text{ integer} \quad \text{for all } e \in E \quad (7)
\end{align*}$$

Let $U$ be any subset consisting of $k$ vertices, where $k \geq 3$ and odd, and let $E(U)$ be the set if edges with both endvertices in $U$. By adding inequalities (5) for all $v \in U$ we obtain

$$\sum_{e \in E(U)} x_e \leq |U|, \quad \text{or equivalently}$$

$$\sum_{e \in E(U)} x_e \leq \frac{|U|}{2}. \quad (8)$$

Since each $x_e$ is an integer, the left-hand side of (8) has to be integral. As $|U|$ is odd, the right-hand side of (8) is fractional, and hence we can round down the right-hand side of (8) giving the valid inequality

$$\sum_{e \in E(U)} x_e \leq \left\lfloor \frac{|U|}{2} \right\rfloor \quad (9)$$

which we call an odd-set constraint. It is easy to show that the odd-set constraints are necessary to describe the convex hull of matchings in $G$. We also note that there are exponentially many odd-set constraints as there are exponentially many ways of forming subsets $U$. We shall now give a more formal description of Chvátal's procedure.
An inequality \( \sum_{j=1}^{n} a_{ij} x_j \leq b_i \) is said to belong to the elementary closure of a set \( P \) of linear inequalities, denoted \( e^1(P) \), if there are inequalities \( \sum_{j=1}^{n} a_{ij} x_j \leq b_i \) \( i = 1, \ldots, m \) in \( P \) and nonnegative real numbers \( \lambda_1, \lambda_2, \ldots, \lambda_m \) such that

\[
\sum_{i=1}^{m} \lambda_i a_{ij} = a_j \text{ with } a_j \text{ integer}, \quad j = 1, \ldots, n,
\]

and

\[
\left\lfloor \sum_{i=1}^{m} \lambda_i b_i \right\rfloor \leq b.
\]

For integer values of \( k > 1 \), \( e^k(P) \) is defined recursively as \( e^k(P) = e(P \cup e^{k-1}(P)) \). The closure of \( P \) is defined as \( c(P) = \bigcup_{k=1}^{\infty} e^k(P) \).

**Theorem 2** Chvátal (1973). If \( S \) is a bounded polyhedron, then \( \text{conv}(S) \) can be obtained after a finite number of closure operations.

An interesting question is if \( k \) can be bounded from above by a function of the dimension of \( S \). Chvátal called the minimum number of closure operations \( k \) required to obtain \( \text{conv}(S) \), given a linear formulation \( P \), the rank of \( P \). If we return to the matching problem (5)–(7), it was proved by Edmonds (1965) that the convex hull of the matching polytope is determined by inequalities (5), (6) and (9). As the odd-set constraints (9) can be obtained by applying one closure operation on the linear formulation, the rank of the set of inequalities (5) and (6) is one. In general however, there is no upper bound on \( k \) in terms of the dimension of \( S \) as the following two-dimensional problem illustrates.

\[
\begin{align*}
\text{Max} & \quad x_2 \\
0 \leq x_1 & \leq 1 \\
x_2 & \geq 0 \\
-tx_1 + x_2 & \leq 1 \\
-tx_1 + x_2 & \leq t + 1 \\
1, x_2 & \text{ integer.}
\end{align*}
\]

Only if \( S = \emptyset \) there exists an upper bound on \( k \) that is a function of the dimension of \( P \). This was proved by Cook et al. (1987).

There is a clear relation between Chvátal’s closure operations and Gomory’s cutting planes in the sense that every Gomory cutting plane can be obtain by a series of closure operations and every inequality belonging to the elementary closure can be obtained as a Gomory cutting plane. It would be possible to prove Theorem 2 by using Gomory’s algorithm, but then one would first need to get rid of the inequalities \( x_j \geq 0, j, \ldots, n \) and the assumption that the entries of \( A \) and \( b \) have to be integer. For further details, see Chvátal (1973).

Schrijver (1980) studied the version of ILP where \( S \) is not necessarily bounded, and where \( P \) is defined by a rational system of linear inequalities. The operations carried out on \( P \) to obtain the convex hull of feasible solutions is quite different from the linear combination and rounding schemes developed by Gomory and Chvátal. The key component of Schrijver’s procedure is the formulation of a totally dual integral (TDI) system of inequalities. A rational system \( Ax \leq b \).
of linear inequalities is TDI if for all integer vectors \( c \) such that \( \max \{ cx : Ax \leq b \} \) is finite, the dual \( \min \{ yb : yA = c, \ y \geq 0 \} \) has an integer optimal solution. Note that if \( Ax \leq b \) is TDI, and if \( b \) is integral, then \( P = \{ x : Ax \leq b \} \) is an integral polyhedron, i.e. all extreme points of \( P \) are integral. TDI systems were introduced by Edmonds and Giles (1977).

Each iteration of Schrijver's procedure consists of the following two steps.

1. Given a rational polyhedron \( P \), find a TDI system \( Ax \leq b \) defining \( P \), with \( A \) integral.

2. Round down the righthand side \( b \).

It has been proved by Giles and Pulleyblank (1979) and Schrijver (1981) that there exists a TDI system as in 1. for every rational polyhedron \( P \), and that the TDI system is unique if \( P \) is full-dimensional. Finding such a TDI system can be done in finite time. After one iteration of the above procedure we get a polyhedron \( P^{(1)} \) strictly contained in \( P \) unless \( P \) is integral. Given the polyhedron \( P^{(1)} \) we repeat the steps 1. and 2. This continues until \( \text{conv}(S) \) is obtained.

**Theorem 3** Schrijver (1980). For each rational polyhedron \( P \), there exists a number \( k \), such that after \( k \) iterations of Schrijver’s procedure \( \text{conv}(S) \) is obtained.

The results presented above are of significant theoretical importance as they give algorithmic ways of generating the convex hull of feasible solutions. All three approaches are finite, but from a practical point of view finite in most cases does not imply that computations can be done within reasonable time. One apparent question is whether for some problem classes it is possible to write down the linear description of the convex hull in terms of concise families of linear inequalities. If that is possible we could apply linear programming directly. This is the topic of the following subsection.

### 1.2 Concise Linear Descriptions

We mentioned in the previous subsection that the convex hull of matchings in a general undirected graph \( G \) is given by the defining inequalities (5), (6) and the exponential class of inequalities (9). Assume now that \( G \) is bipartite, i.e. that we can partition the set \( V \) of vertices into two sets \( V_1, V_2 \) such that all edges have one endpoint in \( V_1 \) and the other endpoint in \( V_2 \). For bipartite graphs the convex hull of matchings is described by the defining inequalities (5) and (6) alone which is a polynomial system of linear inequalities. This means that for bipartite graphs the integrality condition (7) is redundant. In contrast, there is no concise linear description known for the traveling salesman problem, even if we allow for exponential families of inequalities. The reason why the bipartite matching problem is so easy is that the constraint matrix is totally unimodular (TU). A matrix \( A \) is TU if each subdeterminant of \( A \) is equal to 0,1 or -1.

**Theorem 4** If \( A \) is a TU matrix the polyhedron \( P = \{ x : Ax \leq b \} \) is integral for all integer vectors \( b \) for which \( P \) is not empty.

Seymour (1980) provided a complete characterization of TU matrices yielding a polynomial algorithm for testing whether a matrix is TU. For a thorough discussion on TU matrices we refer to Schrijver (1986), and Nemhauser and Wolsey (1988).
An observation that is interesting to make in this context is that the bipartite matching problem is polynomially solvable as its linear description is polynomial in the dimension of the problem. For the matching problem in general undirected graphs there is a polynomial combinatorial algorithm due to Edmonds (1965), but the traveling salesman problem is known to be NP-hard. The following theorem confirms that there is a natural link between the computational complexity of a class of problems and the possibility of providing concise linear descriptions of the convex hull of feasible solutions. Before stating the result we need to introduce the following problems:

The lower-bound feasibility problem. An instance is given by integers $m, n$ an $m \times n$ matrix $A$, vectors $b$ and $c$ and a scalar $\delta$. The question is: $\exists x \in \mathbb{Z}^n : Ax \leq b, \ cx > \delta$?

The facet validity problem. An instance is given by the same input as for the lower-bound feasibility problem. The question is: Does $cx \leq \delta$ define a facet of $\text{conv} \{z \in \mathbb{Z}^n : Ax \leq b\}$?

Note that if the lower-bound feasibility problem for a family of polyhedra is NP-complete then optimizing over the same family of polyhedra is NP-hard.

**Lemma 5** If any NP-complete problem belongs to co-NP, then $NP=co-NP$.

**Theorem 6** Karp and Papadimitriou (1980). If lower-bound feasibility is NP-complete, and facet validity belongs to $NP$ then $NP=co-NP$.

The way to prove Theorem 6 is to show that if facet validity belongs to NP, then lower-bound feasibility belongs to co-NP. Since lower-bound feasibility is NP-complete we can through Lemma 5 conclude that $NP=co-NP$. It is extremely unlikely that $NP=co-NP$, as this implies that all NP-complete problems have a compact certificate for the no-answer. Hence, if we believe that $NP\neq co-NP$, and if $\min \{cx : x \in S\}$ is NP-hard then there are classes of facets of $\text{conv}(S)$ for which there is no short proof that they are facets.

### 1.3 Equivalence Between Optimization and Separation

We have seen that if a problem is NP-hard we cannot expect to have a concise linear description of the convex hull of feasible solution. Moreover, for the matching problem, which is polynomially solvable and which has a concise linear description of the convex hull of feasible solution, this description is exponential in the dimension of the problem. These observations do not necessarily have to be bad news since what we primarily need is a good description of the area around the optimal solution. The question then is whether it is possible to identify a violated inequality whenever needed, i.e. if we can find a hyperplane separating a given fractional solution from the convex hull, or prove that no such hyperplane exists.

The separation problem for a family $FP$ of polyhedra. Given a polyhedron $P \in FP$, and a solution $x^*$, find an inequality $cx \leq \delta$, valid for $P$, satisfying $cx^* > \delta$, or prove that $x^* \in P$.

The optimization problem for a family $FP$ of polyhedra. Given is a polyhedron $P \in FP$. Assume that $P \neq \emptyset$ and that $P$ is bounded. Given a vector $c \in \mathbb{R}^n$, find a solution $x^0$ such that $cx^0 \leq cx$ for all $x \in P$. 

Theorem 7 Grötschel, Lovász and Schrijver (1981). There exists a polynomial time algorithm for the separation problem for a family $F_P$ of polyhedra, if and only if there exists a polynomial time algorithm for the optimization problem for $F_P$.

The theorem says that separation in general is equally hard as optimization but, as we shall see in the next section, when applying the polyhedral approach we develop specific families of valid inequalities for a given problem type, such as the odd-set constraints (9) developed for the matching problem.

The separation problem based on a family $FI$ of valid inequalities. Given a solution $x^*$, find an inequality $cx \leq \delta$ belonging to $FI$, satisfying $cx^* > \delta$, or prove that no such inequality in $FI$ exists.

The separation problem based on a family of valid inequalities may be polynomially solvable even if the underlying optimization problem is NP-hard. Moreover, even if a family of inequalities is NP-hard to separate we may still be able to separate it effectively using a heuristic. Good separation heuristics together with a good implementation of a preprocessing routine and a branch and bound scheme, form the basis for the success of the polyhedral approach.

2 Polyhedral Results for Selected Combinatorial Structures

The results presented in the previous section did provide very important theoretical answers, but no efficient computational tools. In the early seventies there was a renewed interest in developing general purpose integer programming solvers. Instead of Gomory’s cutting plane method, which tended to be very time consuming, one developed facet defining inequalities and separation algorithms for various problem types and embedded the separation algorithms in a branch and bound framework. Since the added inequalities could be proved to be necessary to describe the convex hull of feasible solutions one could expect that they would be more effective than the Gomory cutting planes. Moreover, by developing facet defining inequalities and associated separation algorithms for some basic combinatorial structures that occur frequently in more general combinatorial optimization problems, and by implementing these algorithms in commercial software, it would possibly be very useful when solving a wide range of combinatorial problems. In the late seventies and in the eighties remarkable computational progress was made. Here we shall describe some classes of facet defining valid inequalities developed for some basic, important combinatorial optimization problems. The main purpose with the survey is to give an impression of how inequalities and separation algorithms are developed, and how they can be used, not only for the problem for which they are developed, but also for more general structures. After each problem class we shall give references to related work. Since the space provided here is not enough for a complete survey, we recommend the following literature to the interested reader. The books by Schrijver (1986), and Nemhauser and Wolsey (1988) provide a broad theoretical foundation as well as many examples. The article by Jünger et al. (1994) contains a comprehensive survey of computational results obtained by using polyhedral techniques. The last developments on solving large traveling salesman problems is found in the article by Applegate et al. (1994). It should also be noted that Balas et al. (1994) has made a recent computational study of Gomory’s cutting plane algorithm and report encouraging results.
2.1 Preliminaries

In this subsection we introduce probably well-known algebraic subjects like linear and affine spaces. Moreover, the most important aspects of polyhedra, like faces and facets are defined.

The set of linear combinations of a set of vectors \( x^1 \ldots x^K \subset \mathbb{R}^n \) is the linear space \( LS = \{ \sum_{k=1}^{K} \alpha_k x^k | \alpha \in \mathbb{R}^K \} \). If \( x^1 \ldots x^K \) form a minimal system, i.e., none of the vectors is a linear combination of the others, then the vectors \( x^1 \ldots x^K \) are called linearly independent. Equivalently, the vectors \( x^1 \ldots x^K \) are linearly independent if \( \alpha = 0 \) is the unique solution of the system \( \sum_{k=0}^{K} \alpha_k x^k = 0 \). The dimension of a linear space \( LS \), denoted by \( \dim(LS) \) is defined as the minimum number of linearly independent points in the space.

The set of affine combinations of the \( K + 1 \) points \( x^0, x^1 \ldots x^K \subset \mathbb{R}^n \) is called an affine space \( AS = \{ \sum_{k=0}^{K} \alpha_k x^k | \alpha \in \mathbb{R}^{K+1}; \sum_{k=0}^{K} \alpha_k = 1 \} \). Thus, an affine space can be viewed as a linear space translated over a vector \( x^0 \): \( AS = \{ x^0 + \sum_{k=1}^{K} \beta_k (x^k - x^0) | \beta \in \mathbb{R}^K \} \). Hyperplanes in \( \mathbb{R}^n \) are affine spaces. If the set of points \( x^0 \ldots x^K \) is a minimal system, i.e., none of the points is an affine combination of the others, then the points \( x^0 \ldots x^K \) are called affinely independent. Equivalently, the points \( x^0 \ldots x^K \) are affinely independent if \( \alpha = 0 \) is the unique solution of the system \( \sum_{k=0}^{K} \alpha_k x^k = 0; \sum_{k=0}^{K} \alpha_k = 0 \). The dimension of an affine space, denoted by \( \dim(AS) \) is the number of affinely independent points MINUS 1. Thus, if the points \( x^0 \ldots x^K \) are affinely independent, the affine space defined by these points has dimension \( K \).

A polyhedron \( P \) is the set of points satisfying a system of linear constraints, i.e., \( P = \{ x \in \mathbb{R}^n | Ax \leq b \} \). The dimension of \( P \), denoted by \( \dim(P) \), is the dimension of the smallest affine space containing \( P \).

An inequality \( \pi x \leq \pi_0 \) is called valid with respect to \( P \) if each point in \( P \) satisfies the inequality. A valid inequality defines a face \( F \subseteq P \), where \( F \) is the subset of \( P \) that satisfies the valid inequality at equality, i.e., \( F = \{ x \in P | \pi x = \pi_0 \} \). Note that \( F \) is a polyhedron itself. It is said to be proper if it is not empty and if it is properly contained in \( P \), i.e., \( \emptyset \neq F \neq P \). The dimension of a proper face \( F \), \( \dim(F) \), is strictly smaller than the dimension of \( P \). If \( \dim(F) = \dim(P) - 1 \), i.e., if \( F \) is maximal, then \( F \) is called a facet. The importance of facet-defining inequalities stems from the fact that these are the unique inequalities, among those that are not satisfied at equality for all points in \( P \), for which a representative is necessary in the description of \( P \), see Nemhauser and Wolsey (1988).

2.2 The Vertex Packing Problem

In the vertex packing problem we want to find a maximum cardinality subset \( V' \) of vertices in an undirected graph \( G = (V, E) \), such that no two vertices in \( V' \) are adjacent. The vertex packing problem is sometimes referred to as the independent set problem or as the stable set problem. Let \( x_v = 1 \) if \( v \in V' \) and let \( x_v = 0 \) otherwise. The integer programming formulation of the vertex packing problem is given below.

\[
\max \sum_{v \in V} x_v \tag{10}
\]

s.t.
\[
x_v + x_w \leq 1 \quad \text{for all } \{v, w\} \in E \tag{11}
\]
Let $X_{VP}$ be the set of feasible solutions to the vertex packing problem in the graph $G$ and let $\alpha(G)$ be the maximum cardinality of a vertex packing in $G$. An edge is called critical if its removal from $G$ produces a graph $G'$ with $\alpha(G') > \alpha(G)$. Chvátal (1975) derived the following general sufficient condition for an inequality to define a facet of $\text{conv}(X_{VP})$.

**Theorem 8** Chvátal (1975). Let $E^*$ be the set of critical edges of $G$. If the graph $G^* = (V, E^*)$ is connected, then the inequality $\sum_{j \in V} x_j \leq \alpha(G)$ defines a facet of $\text{conv}(X_{VP})$.

A clique in a graph $G$ is a complete subgraph of $G$. Since no two vertices in $V'$ are allowed to be adjacent we could take any clique $C$ in $G$ and require that at most one vertex belonging to $C$ should belong to the vertex packing $V'$ giving the valid inequality

$$\sum_{j \in C} x_j \leq 1.$$  \tag{13}

**Theorem 9** Padberg (1973). Let $C$ be a clique in the graph $G$. The inequality (13) defines a facet of $\text{conv}(X_{VP})$ if and only if $C$ is maximal.

**Proof.** The dimension of the vertex packing polytope is $|V|$. Hence, to prove that (13) defines a facet of $\text{conv}(X_{VP})$ we need to find $|V|$ affinely independent points that are tight for (13). Let $C$ be a maximal clique. The following vertex packings induce $|V|$ tight affinely independent characteristic vectors. For $v \in C$ we take the vertex packing that contains only $v$. For $v \notin C$ we first choose a node $w \in C$ that is not connected by an edge with $v$. Because of the maximality of $C$ such a node exists. We take the vertex packing that contains both nodes $v$ and $w$. The corresponding characteristic vectors all satisfy the clique constraint at equality. Thus, the constraint is facet-defining.

If $C$ is not maximal, i.e., there is a clique $C'$ such that $C \subseteq C'$, then the clique constraint defined by $C'$ dominates the constraint defined by $C$. \hfill \Box

Another class of valid inequalities for the vertex packing problem is the family of odd-hole inequalities. An odd hole $H$ in a graph $G$ is a chordless cycle consisting of an odd number of vertices, i.e. there are no edges of $G$ connecting any nonconsecutive vertices in $H$. Since the number of vertices in $H$ is odd, at most $\lfloor |H|/2 \rfloor = (|H| - 1)/2$ vertices in $H$ can belong to any vertex packing. Hence the following odd-hole inequality is valid,

$$\sum_{v \in H} x_v \leq \frac{|H| - 1}{2}. \tag{14}$$

Padberg showed that (14) defines a facet of $\text{conv}(X_{VP} \cap \{x_j = 0 \text{ for all } j \notin H\})$. In general (14) is not facet-defining, i.e., it represents a face of $\text{conv}(X_{VP})$ of dimension less than $\dim((X_{VP}) - 1$. The question is whether it is possible to increase the dimension of (14) such
that (14) becomes a facet for $\text{conv}(X_{VPG})$. One way of increasing the dimension of a face is through *sequential lifting* (Padberg (1973) and Wolsey (1976)), which is illustrated in the following example.

The inequality $x_1 + x_2 + x_3 + x_4 + x_5 \leq 2$ defines a facet of $\text{conv}(X_{VPG} \cap \{x_6 = 0\})$. The question is whether there exists a constant $\alpha \geq 0$ such that $x_1 + x_2 + x_3 + x_4 + x_5 + \alpha x_6 \leq 2$ defines a facet of $\text{conv}(X_{VPG})$. If $x_6 = 0$, $\alpha$ can take any value, hence assume that $x_6 = 1$. If $x_6 = 1$ we must have $x_j = 0$, $j = 1, \ldots, 5$ since $x_6$ is adjacent to all other vertices. The maximal value of $\alpha$, such that the inequality remains valid, is $\alpha = 2$. In this example we had only one variable set to a fixed value, but in general we include one variable at the time, with a nonnegative coefficient, in the inequality. Theorems 10 and 11 imply that if the inequality is facet defining in the reduced space, and if we “lift” in all variables sequentially with maximal coefficients, then the resulting inequality defines a facet in the full space. Sequential lifting is sequence dependent, i.e. different lifting sequences give rise to different inequalities. Zemel (1978) proposed an alternative lifting procedure, called *simultaneous lifting*. As the name indicates, the coefficients of all variables that are to be lifted are considered simultaneously, yielding inequalities that cannot be obtained, in general, by sequential lifting. For more details on lifting procedures, see also Nemhauser and Wolsey (1988).

The separation problem for clique inequalities consists of finding a maximum weight clique in a graph. This problem is $NP$-hard, and therefore we usually turn to heuristics for finding violated clique inequalities. The separation problem for odd-holes can be solved in polynomial time by a shortest path algorithm on a slightly adapted graph (Hoffman and Padberg (1993)).
Theorem 10 Let $S \subseteq \{0, 1\}^n$. Suppose $\sum_{j=2}^{n} \pi_j x_j \leq \pi_0$ is valid for $S \cap \{x \in \{0, 1\}^n | x_1 = 0\}$ Then $\alpha x_1 + \sum_{j=2}^{n} \pi_j x_j \leq \pi_0$ is valid if $\alpha \leq \pi_0 - \max_{S \cap \{x \in \{0, 1\}^n | x_1 = 1\}} \{\sum_{j=2}^{n} \pi_j x_j\}$

The dimension of the face represented by the inequality increases by one if $\alpha$ is chosen maximal.

Proof. The validity of the inequality is immediate. Thus, it remains to show that the number of affinely independent vectors satisfying the constraint at equality increases by one for $\alpha$ at its maximum value. Since $\alpha$ is chosen maximal there is a solution with $x_1 = 1$ satisfying the constraint at equality. This solution forms together with a set of affinely independent solutions with $x_1 = 0$ an affinely independent system. Therefore, the dimension of the face represented by the inequality increases by one.

Theorem 11 Let $S \subseteq \{0, 1\}^n$. Let $\sum_{j=2}^{n} \pi_j x_j \leq \pi_0$ is valid for $S \cap \{x \in \{0, 1\}^n | x_1 = 1\}$ Then $\beta_1 x_1 + \sum_{j=2}^{n} \pi_j x_j \leq \pi_0 + \beta_1$ is valid if $\beta_1 \geq \max_{S \cap \{x \in \{0, 1\}^n | x_1 = 0\}} \sum_{j=2}^{n} \pi_j x_j - \pi_0$

The dimension of the face represented by the inequality increases by one if $\beta_1$ is chosen minimal.

Proof. The second theorem follows from the first by multiplying the constraint by $-1$, and then using the complement $x_1' = 1 - x_1$ of the variable $x_1$.

2.3 The Traveling Salesman Problem

In the traveling salesman problem (TSP) one is asked to determine a cycle that contains each vertex exactly once, i.e., a Hamiltonian cycle, of shortest distance in an undirected complete graph $G = (V, E)$ with $n = |V|$ cities. The distances are denoted by $d_e$, for all $e \in E$. The problem is formulated using the variables $x_e (e \in E)$, where $x_e = 1$ if $e$ is chosen in the cycle, and $x_e = 0$ otherwise. Usually, the vertices of the graph are called cities and the Hamiltonian cycle is called a tour.

$$\min \sum_{e \in E} d_e x_e \quad \text{(15)}$$

s.t.  

$$\sum_{e : v \in e} x_e = 2 \quad \text{for all } v \in V \quad \text{(16)}$$

$$\sum_{e \in S} x_e \leq |S| - 1 \quad \text{for all } S : \emptyset \neq S \neq V \quad \text{(17)}$$

$$x_e \in \{0, 1\} \quad \text{for all } e \in E \quad \text{(18)}$$

The formulation restricted to the constraints (16) and (18) allows for solutions consisting of disjoint circuits (subtours), so-called 2-matchings. The constraints (17) prevent these solutions,
and are therefore called the subtour elimination constraints (SEC). They were introduced by Dantzig et al. (1954). By removing the SEC's from the formulation above, we obtain the 2-matching relaxation. Edmonds (1965) studied the facets of the 2-matching problem, and obtained a complete linear description of this problem by adding new constraints, the 2-matching inequalities (named after the problem), which are also valid for the TSP. We introduce the 2-matching constraints with the following example.

![Figure 4: A fractional solution violating a 2-matching constraint](image)

The thick lines correspond to variables that have value 1 and the thin lines to variables with value 0.5. Clearly, this solution satisfies the degree constraints (and the SEC's). To cut off this solution from the convex hull of 2-matchings we introduce the following constraint.

Consider the set of vertices \( H = \{1, 2, 3\} \). Let \( E(H) \) be the set of edges with both endvertices in \( H \), and let \( E' = \{\{1, 4\}, \{2, 5\}, \{3, 6\}\} \), i.e., each edge of \( E' \) has exactly one endvertex in \( H \). From the set of edges \( E(H) \cup E' \) we can at most take four of them in a 2-matching. Otherwise, at most one edge can be deleted from the set \( E(H) \cup E' \), and thus at least one of the vertices in \( H \) will have degree 3, which is not allowed in a tour. The cumulative value of the variables of these edges is 4.5. Defining \( x(F) = \sum_{e \in F} x_e \), we can conclude that the inequality \( x(E(H)) + x(E') \leq 4 \) is violated.

In general, a 2-matching constraint has the form

\[
x(E(H)) + x(E') \leq |H| + \frac{1}{2}|E'|
\]

where \( H \subset V \) and \( E(H) \) the set of edges with both endvertices in \( H \). The edges in \( E' \) have one endvertex in \( H \), i.e., for each \( e \in E' : |e \cap H| = 1 \). Only 2-matching constraints with an odd number of edges in \( E' \), can be facet-defining, since the remaining inequalities are implied by the degree constraints. Comb-inequalities were introduced by Chvátal (1975) as a generalization of the 2-matching constraints. In these inequalities the edges in \( E' \) are replaced by an odd number \( s \) of disjoint sets \( T_1, \ldots, T_s \), the teeth, each having at least one vertex in common with the handle \( H \). We give an example of a violated comb inequality below.

\[
x(E(H)) + \sum_{j=1}^{s} x(E(T_j)) \leq |H| + \sum_{j=1}^{s} (|T_j| - 1) - \frac{1}{2}(s + 1) \quad (19)
\]
The following instance shows a fractional solution that satisfies the 2-matching constraints and the subtour elimination constraints, but not the comb inequality defined by \( H = \{1, 5, 6, 7\} \), and \( T_1 = \{1, 2\}, T_2 = \{3, 4, 5, 6\}, T_3 = \{7, 8\} \).

The comb-inequalities were generalized by Grötschel and Padberg (1979). Grötschel and Pulleyblank (1986) introduced clique tree inequalities as a further generalization of combs. Clique trees contain more handles, which are connected through the teeth. The search for other classes of valid inequalities is still vivid. Many exotic classes have been described to date. A good overview of the current state-of-the-art is provided by Applegate et al. (1994). Goemans (1993) considers the quality of the various inequalities with respect to their induced relaxations.

The separation algorithm for SEC’s solves a minimum cut problem, which is polynomial using max-flow algorithms. Separation of the 2-matching constraints is also polynomial (Padberg and Grötschel (1985)), However, violated 2-matching constraints are usually identified heuristically, since this is faster in practice. No exact polynomial time algorithm is known to date, for solving the separation problem of the comb inequalities. However, there are fast heuristic methods that perform quite well. For clique-tree inequalities, in general, there are even no good heuristics to separate them.

Dantzig et al. (1954) and (1958) used cutting planes to solve the TSP. Their initial LP consisted of the constraints (16) and the relaxed (18). The SEC’s were added when violated, by hand. They solved the famous 49-city problem consisting of most capital cities of the states of the USA, by use of seven SEC’s and two other constraints, see Applegate et al. (1994). We provide detailed computational results in sections 3 and 4.

2.4 The Knapsack Problem

Let \( N = \{1, \ldots, n\} \). The knapsack problem is formulated as

\[
\max \sum_{j \in N} c_j x_j
\]  

(20)
\[
\begin{align*}
\text{s.t.} \quad \sum_{j \in N} a_j x_j & \leq b \\
x_j & \in \{0, 1\} \quad \text{for all } j \in N
\end{align*}
\]  

(21)

Assume that the vectors \(a, c\) and the right-hand side \(b\) are rational, and let \(X_k\) denote the set of feasible solutions to the knapsack problem. We call a set \(C\) a cover or a dependent set with respect to \(N\) if \(\sum_{j \in C} a_j > b\). A cover is minimal if \(\sum_{j \in S} a_j \leq b\) for all \(S \subseteq C\). If we choose all elements from the cover \(C\), it is clear that the right-hand side of (21) is exceeded. Hence, the following knapsack cover inequality (Balas (1975), Hammer et al. (1975) and Wolsey (1975)) is valid,

\[
\sum_{j \in C} x_j \leq |C| - 1.
\]

(23)

A generalization of (23) is given by the family of \((1, k)\)-configuration inequalities. Let \(\bar{C} \subseteq N\), and \(t \in N \setminus \bar{C}\) be such that \(\sum_{j \in \bar{C}} a_j \leq b\) and such that \(Q \cup \{t\}\) is a minimal cover for all \(Q \subseteq \bar{C}\) with \(|Q| = k\). Let \(T(r) \subseteq \bar{C}\) vary over all subsets of cardinality \(r\) of \(\bar{C}\), where \(r\) is an integer satisfying \(k \leq r \leq |\bar{C}|\). The \((1, k)\)-configuration inequality

\[
(r - k + 1)x_t + \sum_{j \in T(r)} x_j \leq r
\]

(24)

is valid for \(\text{conv}(X_K)\), and if \(k = |\bar{C}|\) the cover inequalities (23) are obtained. The \((1, k)\)-configuration inequalities are primarily designed to deal with elements \(j\) of the knapsack having a large coefficient \(a_j\).

In general (23) is not facet defining, but as with the odd-hole inequalities (14) they can be lifted to become facets. One special case of a lifted cover inequality, where all lifting coefficients are equal to zero or one, is obtained by considering the extension \(E(C)\) of a minimal cover \(C\), where \(E(C) = \{k \in N \setminus C : a_k \geq a_j, \text{ for all } j \in C\}\). The inequality \(\sum_{j \in E(C)} x_j \leq |C| - 1\) is valid for \(\text{conv}(X_K)\) and under certain conditions it also defines a facet of \(\text{conv}(X_K)\). The most general form of the knapsack cover inequality is obtained by partitioning the set \(N\) in the sets \(\{N', N \setminus N'\}\). Let \(x_j = 0\) for all \(j \in N \setminus N'\), and let \(C'\) be a minimal cover with respect to \(N'\). Moreover, let \(x_j = 1\) for all \(j \in N' \setminus C'\). By using lifting results we can conclude that \(\text{conv}(X_K)\) has a facet of the following form

\[
\sum_{j \in N \setminus N'} \alpha_j x_j + \sum_{j \in N' \setminus C'} \beta_j x_j + \sum_{j \in C'} x_j \leq |C'| - 1 + \sum_{j \in N' \setminus C'} \beta_j.
\]

(25)

Balas (1975) characterized the lifting coefficients \(\alpha_j\) in the case where \(N' \setminus C' = \emptyset\).

The separation problem for the cover inequalities can be viewed as a knapsack problem again as we will show. Let the point \(x^* \in \mathbb{R}^n\) be given. Is there a cover-inequality that is violated by \(x^*\)? Thus:

Is there a \(C \subseteq N\) such that \(\sum_{j \in C} x_j^* > |C| - 1\) and \(\sum_{j \in C} a_j > b\).
We introduce binary variables $z_j$ ($j \in N$) that determine whether the elements are chosen ($z_j = 1$) in the cover or not ($z_j = 0$). In order to obtain a violated cover inequality $z$ should satisfy the following constraints.

$$\sum_{j=1}^{n} x_j^* z_j > \left( \sum_{j=1}^{n} z_j \right) - 1 \quad \text{and} \quad \sum_{j=1}^{n} a_j z_j \geq b + 1$$

besides the integrality of the $z$-variables. Reformulating the first constraint we get

$$\sum_{j=1}^{n} (1 - x_j^*) z_j < 1 \quad \text{and} \quad \sum_{j=1}^{n} a_j z_j \geq b + 1.$$

Thus, to find the most violated cover, one has to solve the following minimization problem.

$$\min \quad \eta = \sum_{j=1}^{n} (1 - x_j^*) z_j$$

subject to

$$\sum_{j=1}^{n} a_j z_j \geq b + 1$$

$$z_j \in \{0, 1\} \quad \text{for all} \ j \in N \quad (28)$$

If $\eta < 1$ then a cover, violated by $x^*$, is found. By complementing the $z$-variables, i.e., by replacing $z_j$ by $1 - z_j^*$ for all $j \in N$, the problem translates into a knapsack problem again. However, this problem is much smaller and therefore easier to solve for two reasons. First, the items $j$ with $x_j^* = 1$ have coefficient 0 in the objective function, so that we can put $z_j = 1$. Second, the items $j$ with $x_j^* = 0$ have coefficient 1 in the objective, and therefore, these items can not be part of a violated cover inequality. Thus, the only interesting items have fractional corresponding variables. The number of such items is usually very small. Crowder et al. (1983) developed a heuristic for the separation problem (based on solving the above optimization problem), and for choosing the sets $N'$ and $C'$. Once a minimal cover $C'$ is generated it is also used in a heuristic for finding a violated $(1, k)$-configuration inequality. They implemented the algorithms and solved large 0-1 integer programming problems by automatically generating knapsack cover inequalities. Recent work on the knapsack polytope is done by Weismantel (1994).

### 2.5 The Single-Node Flow Problem

Consider a single node in a directed graph, and let $N$ be the set of arcs entering the node. The outflow from the node is equal to $b$. Let $x_j$ be a continuous variable denoting the flow on arc $j$, and let $m_j$ be the capacity of arc $j$. If arc $j$ is open, then $y_j = 1$, otherwise $y_j = 0$. The following fixed charge single-node flow structure is a relaxation of many combinatorial flow models,

$$\sum_{j \in N} x_j = b \quad (29)$$
0 ≤ x_j ≤ m_j y_j \quad \text{for all } j \in N \quad (30)

y_j \in \{0,1\} \quad \text{for all } j \in N \quad (31)

Let \( X_{FC} \) denote the set of feasible solutions to (29)--(31). A subset \( J \subseteq N \) is called a flow cover if \( \sum_{j \in J} m_j = b + \lambda \) with \( \lambda > 0 \). If we have a cover \( J \) and if we close one arc \( k \in J \) then
\[
\max\{x_j : j \in J \setminus k\} = \min\{b, \sum_{j \in J \setminus k} m_j\} = \min\{b, b - (m_k - \lambda)\} = b - (m_k - \lambda)^+ \quad \text{yielding the valid inequality}
\]
\[
\sum_{j \in J} x_j \leq b - \sum_{j \in J} (m_j - \lambda)^+(1 - y_j). \quad (32)
\]

**Theorem 12** Padberg, Van Roy and Wolsey (1985). The flow cover inequality (32) defines a facet of \( \text{conv}(X_{FC}) \) if and only if \( \max_{j \in J} m_j > \lambda \).

Let \( z_j = 1 \) if \( j \in J \) and let \( z_j = 0 \) otherwise, and let \( (x_j^*, y_j^*) \) denote a fractional solution. For a given value of \( \lambda \), the separation problem based on the family of flow cover inequalities (32) is formulated as follows.
\[
\max \sum_{j \in N} [x_j^* + (m_j - \lambda)^+(1 - y_j^*)] z_j
\]
\[
\text{s.t. } \sum_{j \in N} m_j z_j = b + \lambda
\]
\[
z_j \in \{0,1\} \quad \text{for all } j \in N \quad (33)
\]

Once we have a set \( J \) satisfying the condition of Theorem 12 we can extend the flow cover inequality by including flow from the arcs belonging to the set \( L \subseteq (N \setminus J) \).

Let \( \bar{m}_l = \max\{\max_{j \in J} \{m_j\}, m_l\} \) for all \( l \in L \). The following extended flow cover inequality is valid for \( \text{conv}(X_{FC}) \),
\[
\sum_{j \in J \cup L} x_j \leq b - \sum_{j \in J} (m_j - \lambda)^+(1 - y_j) + \sum_{j \in L} (\bar{m}_j - \lambda) y_j \quad (35)
\]

Figure 6: single node flow
Padberg et al. (1985) gave sufficient conditions for the extended flow cover inequality to define a facet of $\text{conv}(X_{FC})$. Aardal et al. (1993) showed that the separation problem based on the family of extended flow cover inequalities can be solved in polynomial time if $m_j = m$ for all $j \in N$.

Van Roy and Wolsey (1986) also studied the single-node flow model with both inflow and outflow fixed charge arcs as well as general uncapacitated fixed charge structures, for which they developed various families of facet defining valid inequalities. Separation heuristics for these inequalities are also discussed by Van Roy and Wolsey (1987).

2.6 An Application: The Facility Location Problem

Here we shall discuss how some of the inequalities presented above can be used, and extended, to solve facility location problems. The facility location problem is a well-known combinatorial optimization problem, and is defined as follows. Let $M = \{1, \ldots, m\}$ be the set of facilities, and let $N = \{1, \ldots, n\}$ be the set of clients. Facility $j$ has capacity $m_j$, and client $k$ has demand $d_k$. The total demand of the clients in the set $S \subseteq N$ is denoted by $d(S)$. The fixed cost of opening facility $j$ is equal to $f_j$ and the cost of transporting one unit of goods from facility $j$ to client $k$ is equal to $c_{jk}$.

Let $y_j = 1$ if facility $j$ is open and let $y_j = 0$ otherwise. The flow from facility $j$ to client $k$ is denoted by $v_{jk}$. We want to determine which facility should be opened and how the flow should be distributed between the open facilities and the clients such that the sum of the fixed costs of opening the facilities, and the transportation costs is minimized, and such that all clients are served, and all capacity restrictions are satisfied. The mathematical formulation is given below.

$$\min \left( \sum_{j \in M} f_j y_j + \sum_{j \in M} \sum_{k \in N} c_{jk} v_{jk} \right)$$

subject to:

$$\sum_{j \in M} v_{jk} = d_k \quad \text{for all } k \in N$$

$$\sum_{k \in N} v_{jk} \leq m_j y_j \quad \text{for all } j \in M$$

$$0 \leq v_{jk} \leq d_k y_j \quad \text{for all } j \in M, \ k \in N$$

$$y_j \in \{0, 1\} \quad \text{for all } j \in M$$

2.6.1 The Uncapacitated Case

In the uncapacitated facility location (UFL) problem we have $m_j = d(N)$ for all $j \in M$. It is convenient to scale the flow by substituting the variables $v_{jk}$ by the variables $x_{jk} = v_{jk}/d_k$. The set of feasible solutions to UFL, $X_{UFL}$, is given by the following sets of constraints.

$$\sum_{j \in M} x_{jk} = 1 \quad \text{for all } k \in N$$

$$0 \leq x_{jk} \leq y_j \quad \text{for all } j \in M, \ k \in N$$
It is possible to explicitly require that \( x_{jk} \in \{0, 1\} \) since there is at least one optimal solution of UFL having this property. Moreover, we can change the equality sign in constraint set (40) to a less-than-or-equal-to sign if we make an appropriate change in the objective function. Due to this modification we are ensured that the inequalities (40) will always be satisfied with equality (for more details see Cho et al. (1983)). Finally, by complementing the \( y_j \)-variables, i.e. by introducing \( y'_j = 1 - y_j \), we obtain the following vertex packing formulation of UFL.

\[
\sum_{j \in M} x_{jk} \leq 1 \quad \text{for all } k \in N \tag{43}
\]

\[
x_{jk} + y'_j \leq 1 \quad \text{for all } j \in M, k \in N \tag{44}
\]

\[
y'_j, x_{jk} \in \{0, 1\} \quad \text{for all } j \in M, k \in N \tag{45}
\]

Let \( X_{UFLVP} \) be the set of feasible solutions to (43)–(45). Given a vertex packing formulation of UFL, we can construct an associated undirected graph, called the intersection graph by introducing a vertex for every variable and an edge for every pair of nonorthogonal columns. To determine \( \text{conv}(X_{UFLVP}) \) is equivalent to determining the convex hull of vertex packings in the associated intersection graph. Hence, we can use all results described in Section 2.2 to derive valid inequalities for UFL. Here, all cliques in the intersection graph are described by inequalities (43) and (44), and all odd holes are cycles where every third vertex is a \( y'_j \)-vertex. Both Cornuéjols and Thizy (1982) and Cho et al. (1983) used the result by Chvátal given in Theorem 8 to find more general inequalities than the odd-hole inequalities. All these inequalities have a regular cyclic structure and all coefficients are equal to one for all variables except one example of a simultaneously lifted odd-hole inequality given by Cornuéjols and Thizy. Aardal and Van Hoesel (1995) discuss further use of simultaneous lifting to get new facets having different coefficients.

2.6.2 The Capacitated Case

By aggregating the flow from each depot as well as the demand we can easily see that a version of the knapsack as well as the single node flow structure form relaxations of the capacitated facility location (CFL) problem. Let \( v_j = \sum_{k \in N} v_{jk} \). By using the aggregate flow variable \( v_j \), we can obtain the aggregate capacity and demand constraints

\[
0 \leq v_j \leq m_j y_j \quad \text{for all } j \in M \tag{46}
\]

\[
\sum_{j \in M} v_j = d(N) \quad \text{for all } j \in N \tag{47}
\]

If we combine constraints (46) and (47) with constraint (39) we obtain the single-node flow polytope and the so-called surrogate knapsack polytope \( X_{SK} = \{ y \in \{0, 1\} : \sum_{j \in M} m_j y_j \geq d(N) \} \). Complementing the \( y_j \)-variables, i.e. letting \( y'_j = 1 - y_j \) for all \( j \in M \) gives the knapsack polytope \( \{ y \in \{0, 1\} : \sum_{j \in M} m_j y'_j \leq \sum_{j \in M} m_j - d(N) \} \). Hence we can use both the knapsack
cover inequalities as well as the flow cover inequalities when solving CFL. Both classes of inequalities can also be derived for subsets \( K \subseteq N \) of clients. Especially the cover inequalities have proved very useful computationally, as is discussed further in Section 3. One way of generalizing the flow cover inequalities is by considering a subset of clients as well as subsets of arcs yielding the family of effective capacity inequalities (Aardal et al. (1993)). Let \( K_j \subseteq K \) for all \( j \in M \) and let \( \bar{m}_j = \min \{m_j, d(K_j)\} \). Let \( J \) define a cover with respect to \( K \), i.e. \( \sum_{j \in J} \bar{m}_j = d(K) + \lambda \) with \( \lambda > 0 \). The effective capacity (EC) inequality

\[
\sum_{j \in J} \sum_{k \in K_j} v_{jk} \leq d(K) - \sum_{j \in J} (\bar{m}_j - \lambda)^+(1 - y_j) \quad (48)
\]

is valid for \( \text{conv}(X_{CFL}) \). The facet defining EC inequalities were completely characterized by Aardal et al. (1993). To further generalize the EC inequalities consider the function \( f(J) \) which is the maximum feasible flow from the depots in \( J \) to the clients in \( K \) on the arcs \( \{(j, k) : j \in J, k \in K_j\} \). By using maximum flow arguments we can show that \( (f(J) - f(J \setminus \{j\}) \geq (\bar{m}_j - \lambda)^+ \). Hence the submodular inequality

\[
\sum_{j \in J} \sum_{k \in K_j} v_{jk} \leq f(J) - \sum_{j \in J} (f(J) - f(J \setminus \{j\})(1 - y_j) \quad (49)
\]

is at least as strong as the EC inequality (48). Submodular inequalities were first considered by Wolsey (1989) and further developed for CFL by Aardal et al. (1993). Since there is no closed-form expression of \( f(J) \) in general, it is hard to characterize the submodular facets. Aardal et al. completely characterized two subclasses for which \( (f(J) - f(J \setminus \{j\}) \geq (\bar{m}_j - \lambda)^+ \) for at least one \( j \in J \), namely the single-depot and the multi-depot inequalities. The separation problem based on the EC inequalities and the submodular inequalities are discussed by Aardal (1994).

2.7 A List of Polyhedral Results for Combinatorial Problems

Here we provide a list of polyhedral results that are known for combinatorial optimization problems. If a recent survey of results for a specific problem class is known, we refer to the survey and not to the individual articles. Surveys are marked with an asterisk. Due to the vast literature, we do not claim that the survey is complete.

- **Clique problems**: Pulleyblank and Shepherd (1993), Balas et al. (1994).
- **Clustering**: Grötschel and Wakabayashi (1989).
- **Cut polytopes**: Barahona and Mahjoub (1986), Barahona et al. (1988), Conforti et al. (1990/91a,b), De Sousa and Laurent (1991), Deza et al. (1992), Deza and Laurent (1992a,b).
- **Knapsack problems**: Balas (1975), Hammer et al.

3 Computational aspects

The classes of cutting planes developed in the previous section, and their separation algorithms are used in the following cutting plane algorithm.

**Outline of the cutting plane algorithm**

1. Initialize the linear programming relaxation of the ILP problem.

2. Solve the current linear programming relaxation of the ILP problem. Let $x^*$ be the optimal solution. If $x^*$ is integral stop, otherwise perform step 3.

3. A separation algorithm is run to identify inequalities violated by $x^*$. If one or more inequalities have been found, the process is repeated, starting with step 2. If no violated inequality is found we stop.

If the algorithm ends by finding an integral solution $x^*$, then the problem is provably solved to optimality. Otherwise, if it ends by not finding any violated cuts, the final solution presents us with a lower bound $z^*$ (in case of a minimization problem), on the optimal value of the integer program. Contrary to Gomory's cutting plane algorithm we can not guarantee to find an optimal integer solution always. This follows from the fact that the classes of valid inequalities used for separation do not constitute a complete description of the convex hull of feasible solutions, and moreover we may have to use heuristics to find violated inequalities in the separation algorithm. Nevertheless, it pays to search for strong valid inequalities.

To illustrate the effects on the lower bound of adding constraints sequentially, we consider a TSP instance of 120 cities from Grötschel (1980). This problem has been solved to optimality by use of cutting planes only. This was reached after 13 iterative solves of LP relaxations. In
each iteration a number of violated constraints was added. The values of the respective linear programming relaxations are given in the following table. $z^*$ denotes the value of the optimal solution of the linear program, and $\# \text{ cuts}$ gives the number of valid inequalities found during separation.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$z^*$</th>
<th>$# \text{ cuts}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6,662.5</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>6,883.5</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>6,912.5</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>6,918.8</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>6,928.0</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>6,935.3</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>6,937.2</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>6,939.5</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>6,940.4</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>6,940.8</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>6,941.2</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>6,941.5</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>6,942.0</td>
<td></td>
</tr>
</tbody>
</table>

During the iterations 36 subtour elimination constraints, 25 2-matching constraints, and 35 comb constraints were added. As can be concluded from the table, it is good practice to generate many violated (or at least promising) valid inequalities, and add them to the linear program in one iteration. Empirically, one observes that the computation times of solving the linear programs increase modestly, but the lower bound tends to optimality much quicker, compared to adding the constraints one at a time.

### 3.1 Extensions to the cutting plane algorithm

There are several ways to extend the basic cutting plane algorithm. We will describe the major techniques in the order in which they are processed in the cutting plane algorithm.
### 3.1.1 Preprocessing the linear program

Preprocessing integer linear programs involves removing redundant constraints, and tightening the right-hand side and the variable coefficients of the inequalities. This leads primarily to better lower bounds for the linear programming relaxations, but it may also lead to significant reductions in the size of the formulation, both with respect to the number of constraints and number of variables. There are many preprocessing techniques described in the literature. For each technique, or a combination of techniques, the problem is to find the right balance between effectiveness and computational time. We intend to present the methods that tighten a linear program quickly. Thus, we restrict ourselves to computationally simple tools. Savelsbergh (1994) lists a few methods to do simple preprocessing using the lower and upper bounds of the variables. These methods have been described in Crowder et al. (1983), and Hofmann and Padberg (1991) for binary programs.

Consider the following subset of the constraints from a (mixed) integer program, where \( N \) is the set indices corresponding to variables with nonzero coefficients in 50. \( N^+ \) is the subset of \( N \) corresponding to variables with positive coefficients, and \( N^- \), the subset of \( N \) with negative coefficients, i.e., the coefficients \( a_j \) for all \( j \in N \) are positive in the following inequality.

\[
\sum_{j \in N^+} a_j x_j - \sum_{j \in N^-} a_j x_j \leq b \quad (50)
\]

\[
l_j \leq x_j \leq u_j \quad \text{for all } j \in N \quad (51)
\]

A lower bound on the left-hand side of (50) is \( LB = \sum_{j \in N^+} a_j l_j - \sum_{j \in N^-} a_j u_j \). If \( LB > b \), then the problem is infeasible. An upper bound on the left-hand side of (50) is \( UB = \sum_{j \in N^+} a_j u_j - \sum_{j \in N^-} a_j l_j \). If \( UB \leq b \), then the constraint is redundant. Similarly, one can improve the coefficients of the variables by selecting one variable, removing it from the left-hand side of (50), and repeating the process.

An elegant preprocessing technique is probing on the variables, i.e., fixing a variable temporarily. This may lead to, besides the aforementioned goals, logical constraints among the variables, which can be used to tighten inequalities, and to obtain new, stronger inequalities. The initial paper on probing techniques is Guignard and Spielberg (1981). We illustrate the effects of probing on the following example.

**Example**

Consider the following set of constraints with two binary variables \( x_1 \) and \( x_2 \), and two nonnegative real variables \( y_1 \) and \( y_2 \).

\[
\begin{align*}
y_1 + 3y_2 & \geq 12 \\
2y_1 + y_2 & \geq 15 \\
y_1 & \leq 10x_1 \\
y_2 & \leq 20x_2
\end{align*}
\]

We probe on \( x_1 \) by setting \( x_1 = 0 \). Then also \( y_1 = 0 \), and thus \( y_2 \geq 15 \), and \( x_2 = 1 \). We can add to the right-hand side of the first inequality \( 33(1 - x_1) \) maintaining validity, i.e., we get
33x_1 + y_1 + 3y_2 \geq 45.

Note that in this example \( x_1 = 0 \) implies \( x_2 = 1 \). Thus, we have \( x'_1 + x'_2 \leq 1 \), where \( x'_i \) (\( i = 1, 2 \)) denotes the complementary variable of \( x_i \). Implications derived from relations between binary variables can be used to derive inequalities among these variables. Especially for binary variables such constraints can be tightened using cliques as follows. Draw an auxiliary graph that contains nodes for the binary variables and their complements. Nodes are connected if their corresponding variables can not both have value 1. The following auxiliary graph is constructed from four binary variables.

![Auxiliary Vertex Packing graph.](image)

We can conclude that \( x'_2 = 0 \) and therefore \( x_2 = 1 \). This gives \( x_3 = 0 \) and \( x_4 = 0 \). Thus, \( x'_3 = 1 \), which again leads to \( x'_1 = 0 \). In other words, we may be able to fix variables. In general, the cliques in the auxiliary graph induce inequalities tighter than those in the original formulation.

The effectiveness of the preprocessing techniques has been tested by Savelsbergh (1994) on a set of 10 mixed integer programming problems from the literature. The following table shows the improvement in lower bound after preprocessing the formulation of these problems. Moreover, the number of nodes in the branch and bound tree is given for both the original and the preprocessed formulations.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Original Lower Bound</th>
<th>Preprocessed Lower Bound</th>
<th>Original Nodes</th>
<th>Preprocessed Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Observe that the linear programming bound increases substantially for all problems after preprocessing. The branch-and-bound tree also reduces much in most problems. Nevertheless, there are two problems for which the branch-and-bound tree of the preprocessed problem is larger than the tree of the original problem. This phenomenon is contra-intuitive, and it may just be a coincidence.

Hofmann and Padberg (1991) and Dietrich and Escudero (1990) describe preprocessing techniques similar to the above with substructures other than variable lower and upper bounds.

### 3.1.2 Postprocessing the linear program

After the linear program is solved, a lower bound \( z^* \) on the optimal value of the IP is available, together with the (usually) fractional optimal LP-solution \( x^* \). Suppose that we know a feasible solution of the IP with value \( z^f \). \( z^f \) is an upper bound on the optimal value of the IP which can be guaranteed to be in the interval \([z^*, z^f]\). Heuristics that use the LP solution \( x^* \) to create a
Table 1: Effects of preprocessing techniques

<table>
<thead>
<tr>
<th>Problem</th>
<th>$z_{LP}$ without</th>
<th>$z_{LP}$ with</th>
<th>$z_{MIP}$</th>
<th># B&amp;B nodes without</th>
<th># B&amp;B nodes with</th>
</tr>
</thead>
<tbody>
<tr>
<td>egout</td>
<td>149.5</td>
<td>562.1</td>
<td>568.1</td>
<td>553</td>
<td>3</td>
</tr>
<tr>
<td>fixnet3</td>
<td>40717.0</td>
<td>50414.2</td>
<td>51973.0</td>
<td>131</td>
<td>5</td>
</tr>
<tr>
<td>fixnet4</td>
<td>4257.9</td>
<td>7703.4</td>
<td>8936.0</td>
<td>2561</td>
<td>1031</td>
</tr>
<tr>
<td>fixnet6</td>
<td>1200.8</td>
<td>3192.5</td>
<td>3983.0</td>
<td>4795</td>
<td>4305</td>
</tr>
<tr>
<td>khb05250</td>
<td>95919464.0</td>
<td>106750366.0</td>
<td>106940226.0</td>
<td>11483</td>
<td>13</td>
</tr>
<tr>
<td>gen</td>
<td>112130.0</td>
<td>112271.0</td>
<td>112313.0</td>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>att</td>
<td>125.9</td>
<td>149.1</td>
<td>160.2</td>
<td>6459</td>
<td>127</td>
</tr>
<tr>
<td>sample2</td>
<td>247.0</td>
<td>290.4</td>
<td>375.0</td>
<td>336</td>
<td>51</td>
</tr>
<tr>
<td>p0033</td>
<td>2520.8</td>
<td>2838.5</td>
<td>3089.0</td>
<td>15</td>
<td>7</td>
</tr>
<tr>
<td>lseu</td>
<td>834.6</td>
<td>947.9</td>
<td>1120.0</td>
<td>297</td>
<td>464</td>
</tr>
</tbody>
</table>

feasible solution are known as primal heuristics. Rounding the fractional variables of the LP solution is the simplest way to achieve this. These upper bounds can also be used to fix variables by reduced cost fixing, or more involved, by parametric analysis on the variables.

### 3.1.3 The separation process

Besides the problem specific classes of valid inequalities, we can try to find violated classes of generic inequalities. Many problems contain knapsack like constraints, so that we may be able to find violated (extended) knapsack cover inequalities. Other generic classes of valid inequalities are clique inequalities, obtained from the auxiliary graph of the binary variables, and flow cover inequalities, obtained from variable upper bound constraints. The facility location problem provides a good insight in what these generic inequalities might offer. The table below shows the improvement that is obtained over the value of the initial linear program by use of knapsack covers only.

<table>
<thead>
<tr>
<th>problem</th>
<th>duality gap (%)</th>
<th># B&amp;B nodes</th>
<th>time (s)</th>
<th># cover inequalities</th>
<th>% gap closed</th>
<th># B&amp;B nodes</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25081</td>
<td>5.9</td>
<td>23</td>
<td>8</td>
<td>4</td>
<td>100.0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>25082</td>
<td>10.3</td>
<td>125</td>
<td>34</td>
<td>10</td>
<td>74.3</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>25083</td>
<td>7.5</td>
<td>79</td>
<td>25</td>
<td>6</td>
<td>85.5</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>25084</td>
<td>2.2</td>
<td>9</td>
<td>6</td>
<td>1</td>
<td>100.0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>25085</td>
<td>5.2</td>
<td>19</td>
<td>7</td>
<td>5</td>
<td>86.6</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>50331</td>
<td>1.5</td>
<td>399</td>
<td>686</td>
<td>13</td>
<td>86.0</td>
<td>31</td>
<td>125</td>
</tr>
<tr>
<td>50332</td>
<td>1.2</td>
<td>691</td>
<td>1560</td>
<td>58</td>
<td>54.3</td>
<td>51</td>
<td>450</td>
</tr>
<tr>
<td>50333</td>
<td>1.5</td>
<td>259</td>
<td>556</td>
<td>122</td>
<td>54.1</td>
<td>89</td>
<td>769</td>
</tr>
<tr>
<td>50334</td>
<td>0.7</td>
<td>239</td>
<td>493</td>
<td>42</td>
<td>76.6</td>
<td>23</td>
<td>213</td>
</tr>
<tr>
<td>50335</td>
<td>1.3</td>
<td>685</td>
<td>1232</td>
<td>25</td>
<td>78.3</td>
<td>49</td>
<td>248</td>
</tr>
</tbody>
</table>
3.2 Embedding the cutting-plane algorithm in a branch-and-bound framework

In the early days solving hard problems was done by applying a cutting plane algorithm, followed by a straightforward branching process. At first in the mid-eighties Grötschel et al. (1984) used the cutting plane algorithm in every node of the branching tree for the linear ordering problem. Padberg and Rinaldi (1987) called this idea branch-and-cut, and applied it to the traveling salesman problem.

Outline of the branch-and-cut algorithm

1. Initialize a list $L$ of subproblems, with the original problem. Then repeat the following steps, until $L$ is empty.
2. Select a subproblem $S$ from $L$.
3. Process $S$ with the (extended) cutting plane algorithm.
4. If $S$ is not solved, then branch on (a variable in) $S$. Put the subproblems created in $L$.

The branch and bound framework contains two new features. During the process a set of subproblems (subinstances) is maintained, which are called the active subproblems. From these subproblems we choose one to work on. Well-known choice criteria are depth-first search, breadth-first search, and best-first search. Then the extended cutting plane algorithm can be performed on the subproblem. If this does not lead to the conclusion that the subproblem can be fathomed, either because the solution found is integral or because the optimal value (lower bound) is higher than a known upper bound (the best known feasible solution), then the subproblem is partitioned into new (active) subproblems by some branching rule.

The variable branching rules select a variable, and create a branch for each value this variable can obtain. The most common ways to select (binary) variables are listed here.

1. Select the variable with value closest to 0.5;
2. Select the variable with value closest to 1;
3. Select the variable with highest objective coefficient;
4. Select a set $L$ of promising variables and compute, for each variable the effects for all branches. Then select the variable for which the branch with the smallest lower bound is best.

Padberg and Rinaldi (1991) suggest a combination of (1) and (3) for the traveling salesman problem. Rule (2) is surprisingly effective in combination with a depth-first search strategy. Rule (4) has similarities with the “steepest-edge” idea as implemented in the simplex method for linear programming. It has been introduced by Applegate et al. (1994). Other strategies have been proposed by Balas and Toth (1985). Jünger et al. (1992) provide computational experience with some (combinations) of these rules.
Figure 9: Branch-and-cut algorithm.

Branching rules which select a constraint, usually a clique constraint, create a branch for each value the left-hand side of the constraint can obtain. Clochard and Naddef (1993) suggest such a rule for the traveling salesman problem.

Strategic issues and implementation.

The particular elements of the extended cutting plane algorithm may not be very effective in each node (subproblem) of the branch-and-cut tree. For instance, preprocessing has much effect in the root node of the tree, since the original formulation of a problem usually contains a lot of redundancy. Similarly, in subproblems it may be hard to find effective cutting planes, and therefore usually the major effort on separation is put in the root node (original problem). Actual implementations of branch-and-cut algorithms contain selection mechanisms for the components with respect to the nodes where these components are performed. Effectiveness versus computational effort is the decisive criterion here. Balas et al. (1994) experimented successfully, with Gomory’s cutting planes, selecting them in nodes at specified depths of the search tree.

The use of upper bounds (feasible solutions) has been mentioned already for doing reduced cost fixing. In the branch-and-cut context a good feasible solution is even more vital for fathoming subproblems quickly. Therefore, the branch-and-cut procedure is usually preceded by a state-of-the-art heuristic. If the gap between the lower bound of a subproblem and the value of best known feasible solution is fairly large, this may be due to the latter (traveling salesman problem). As a result few nodes will be fathomed on the basis of their lower bounds. Branch pausing (Padberg and Rinaldi (1991)) is a strategy in which subproblems with high lower bounds are temporarily ignored, if their lower bounds pass a certain threshold. This threshold is an estimate of the optimum value of the problem. A selection mechanism like best-first search automatically handles the subproblems with high lower bounds last. This mechanism, however,
complicates the implementation of the branch-and-cut algorithm in the sense that subsequently chosen subproblems have no relation to each other.

The task of maintaining the cutting planes is rather difficult in specific implementations of branch-and-cut. In early versions of branch-and-cut packages, one was only allowed to generate globally valid cuts, i.e., cuts valid for the original problem instance. These cuts were maintained in a pool, from which one could select promising ones for the subproblem at hand. The global cuts usually work well enough. However, to obtain the full power of the branch-and-cut algorithm, one should be able to generate constraints, which are locally valid only. Balas et al. (1994) report very good results using branch-and-cut with locally valid Gomory cuts. A detailed overview of the implementational ideas, in general, can be found in Jünger et al. (1994). Data structures and other implementational details, specific for the traveling salesman problem can be found in Applegate et al. (1994).

Applications of branch and cut algorithms are numerous. We will report on a few important ones in the section on computational results. Below we present the branch-and-cut tree of a 532-city traveling salesman problem, an instance that was solved by Padberg and Rinaldi (1987). This tree gives an indication of the development of the lower bounds in the nodes of the tree.
Figure 10: Branch-and-cut tree for the 532-city TSP.
4 Computational Results for Selected Problems

In this section we present computational results for the problems mentioned in previous sections, and a few others. They should give an idea of what problem types can be solved by branch-and-cut algorithms, and what the state-of-the-art is for these problems in terms of solvable sizes and computation times.

4.1 General zero-one linear programs

Crowder et al. (1983) present the first computational results for large-scale zero-one linear programs. On a test set of 10 problems they show the effects of simple preprocessing techniques, and general cutting planes (knapsack covers and \((1,k)\)-configurations. They use these techniques in the root node of a branch-and-bound tree. In the other nodes they use only reduced-cost fixing to eliminate variables. Their results on the test set are shown in the following table.

<table>
<thead>
<tr>
<th>Original Problem</th>
<th>After preprocessing</th>
<th>Cutting planes</th>
<th>Branch &amp; Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vars</td>
<td>Rows</td>
<td>(Z_{LP})</td>
<td>Vars</td>
</tr>
<tr>
<td>33</td>
<td>16</td>
<td>2520.6</td>
<td>33</td>
</tr>
<tr>
<td>40</td>
<td>24</td>
<td>61796.5</td>
<td>40</td>
</tr>
<tr>
<td>201</td>
<td>134</td>
<td>6875.0</td>
<td>195</td>
</tr>
<tr>
<td>282</td>
<td>242</td>
<td>176867.5</td>
<td>282</td>
</tr>
<tr>
<td>291</td>
<td>253</td>
<td>1705.1</td>
<td>290</td>
</tr>
<tr>
<td>548</td>
<td>177</td>
<td>315.3</td>
<td>527</td>
</tr>
<tr>
<td>1550</td>
<td>94</td>
<td>1706.5</td>
<td>1550</td>
</tr>
<tr>
<td>1939</td>
<td>109</td>
<td>2051.1</td>
<td>1939</td>
</tr>
<tr>
<td>2655</td>
<td>147</td>
<td>6532.1</td>
<td>2655</td>
</tr>
<tr>
<td>2756</td>
<td>756</td>
<td>2688.7</td>
<td>2734</td>
</tr>
</tbody>
</table>

4.2 The Traveling Salesman problem.

The literature on computational results for the traveling salesman problem is huge. Some of the results have already been shown in previous sections. Several research groups have branch-and-cut codes available. To make the progress visual, we give a list of world-records with respect to the size of the problems solved. It should be noted however, that there are still some small instances unsolved, which indicates that large is not synonymous with difficult. The instances mentioned in the literature are all Euclidean symmetric Traveling Salesman problems. They arise from applications like road maps, routing of drilling machines (in chip-technology), and x-ray crystallography. The instances can be found in the library NETLIB (Reinelt (1991)).

The table below contains information on the number of cities \(n\) of the instance (the number of variables is \(\frac{1}{2}n(n-1)\)), the value of the LP-solution in the root node after the cutting plane phase, the value of the optimal solution, and the number of branch-and-cut nodes. Furthermore, we give side information like the authors that reported the problem solved, the year, and the application. The data are from the original papers, so later techniques may have different performance. To illustrate this, for the 532-instance there are at least three different numbers of branch-and-cut nodes.
<table>
<thead>
<tr>
<th>Cities</th>
<th>LP value at root</th>
<th>IP value</th>
<th># B&amp;C Nodes</th>
<th>Application</th>
<th>Year solved</th>
<th>Solved by</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>12345</td>
<td>12345</td>
<td>1</td>
<td>Roads USA</td>
<td>1954</td>
<td>Dantzig et al.</td>
</tr>
<tr>
<td>120</td>
<td>6942</td>
<td>6942</td>
<td>1</td>
<td>Roads Ger</td>
<td>1980</td>
<td>Grötschel</td>
</tr>
<tr>
<td>318</td>
<td>??</td>
<td>41349</td>
<td>??</td>
<td>Drilling</td>
<td>1980</td>
<td>Crowder &amp; Padberg</td>
</tr>
<tr>
<td>532</td>
<td>27628</td>
<td>27686</td>
<td>85</td>
<td>Roads USA</td>
<td>1987</td>
<td>Padberg &amp; Rinaldi</td>
</tr>
<tr>
<td>666</td>
<td>294050</td>
<td>294358</td>
<td>21</td>
<td>Worldmap</td>
<td>1991</td>
<td>Grötschel &amp; Holland</td>
</tr>
<tr>
<td>1002</td>
<td>258860</td>
<td>259045</td>
<td>13</td>
<td>Drilling</td>
<td>1990</td>
<td>Padberg &amp; Rinaldi</td>
</tr>
<tr>
<td>2392</td>
<td>378027</td>
<td>378032</td>
<td>3</td>
<td>Drilling</td>
<td>1990</td>
<td>Padberg &amp; Rinaldi</td>
</tr>
<tr>
<td>3038</td>
<td>137660</td>
<td>137694</td>
<td>287</td>
<td>Drilling</td>
<td>1992</td>
<td>Applegate et. al</td>
</tr>
<tr>
<td>4461</td>
<td>182528</td>
<td>182566</td>
<td>2092</td>
<td>??</td>
<td>1994</td>
<td>Applegate et. al</td>
</tr>
<tr>
<td>7397</td>
<td>23253123</td>
<td>23260728</td>
<td>2247</td>
<td>??</td>
<td>1994</td>
<td>Applegate et. al</td>
</tr>
</tbody>
</table>

To give an idea of the solution times, the 2392-city problem was solved in appr. 6 hours on a CYBER. As can be seen from the table the lower bounds in the root node are very close to optimum. This explains part of the success of cutting plane algorithms for the symmetric TSP. The other part is due to many computational ideas during the years. For a very good reference on these ideas see Applegate et al. (1994).

4.3 The Vertex Packing problem.

Nemhauser and Sigismundi (1992) report on solving randomly generated instances of the maximum cardinality vertex packing problem with sizes varying from 40 to 120 nodes. For these sizes they vary the density, i.e., the probability that an edge is in the graph, from 0.1 to 0.9. The code of the authors was limited, in the sense that the cutting plane algorithm could only be run in the root node, and primitive branching rules were available. Below we give a table for the 0.2 density problems.

The table gives information of the number of nodes of the graph (Size), the initial gap (in %), the number of clique inequalities and odd-hole constraints, the gap (in %) remaining after the cutting plane phase, the number of branch-and-bound nodes, and the total number of iterations of the simplex method.

<table>
<thead>
<tr>
<th># Nodes</th>
<th>Initial Gap</th>
<th># Clique inequalities</th>
<th># Odd-hole inequalities</th>
<th>Gap after C.P. phase</th>
<th># B&amp;B Nodes</th>
<th># LP Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>7</td>
<td>86</td>
<td>0</td>
<td>1</td>
<td>16</td>
<td>1439</td>
</tr>
<tr>
<td>60</td>
<td>13</td>
<td>203</td>
<td>36</td>
<td>1</td>
<td>97</td>
<td>13352</td>
</tr>
<tr>
<td>80</td>
<td>21</td>
<td>369</td>
<td>33</td>
<td>4</td>
<td>97</td>
<td>13352</td>
</tr>
<tr>
<td>90</td>
<td>15</td>
<td>222</td>
<td>13</td>
<td>2</td>
<td>58</td>
<td>3649</td>
</tr>
<tr>
<td>100</td>
<td>29</td>
<td>181</td>
<td>19</td>
<td>2</td>
<td>108</td>
<td>6631</td>
</tr>
<tr>
<td>110</td>
<td>35</td>
<td>781</td>
<td>5</td>
<td>8</td>
<td>394</td>
<td>84115</td>
</tr>
<tr>
<td>120</td>
<td>40</td>
<td>903</td>
<td>5</td>
<td>11</td>
<td>251</td>
<td>35194</td>
</tr>
</tbody>
</table>

Clique inequalities close most of the gap between the lower bound and the optimal solution. In low-density graphs (lifted) odd-holes are important. Medium density problems are the most difficult instances. In fact, the authors were not able to solve some 120 vertex problems within 100000 LP iterations.
One can conclude that random vertex packing problems are difficult to solve with cutting plane algorithms. However, if we consider structured vertex packing problems, we may be able to solve larger instances. Two applications are considered below. These show that we actually can solve large-scale vertex packing alike problems.

### 4.4 Frequency Assignment

In the frequency assignment problem we are given a set of links that have to be assigned frequencies. These frequencies can be chosen from a set depending on the link. Moreover, the assigned frequencies should satisfy certain distance constraints. The objective is to minimize the number of frequencies used. The modelling of the problem is done by introducing a binary variable for each feasible link-frequency pair. The size of the problem instances is measured in the number of links (each link actually comes in a pair of two connections). The number of binary variables is obtained approximately by multiplying the number of links with 40 (the average number of potential frequencies for the links). Aardal et al. (1995) report on solving the following problems.

<table>
<thead>
<tr>
<th># Links</th>
<th>Initial lower bound</th>
<th>Final lower bound</th>
<th>Best known solution</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>46</td>
</tr>
<tr>
<td>200</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>1925</td>
</tr>
<tr>
<td>340</td>
<td>20</td>
<td>22</td>
<td>22</td>
<td>6167</td>
</tr>
<tr>
<td>458</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>400</td>
</tr>
</tbody>
</table>

### 4.5 The set partitioning problem: airline crew scheduling

Hoffman and Padberg (1993) report on solving huge, in terms of variables, set partitioning problems arising in Airline Crew Scheduling problems. The cutting plane phase uses inequalities similar to those for the vertex packing problem., i.e., clique and (lifted) odd-hole constraints and preprocessing techniques. The branch-and-cut phase contains a variable branching rule. We selected a few instances, the ones with the highest number of rows and the ones with the highest number of columns.

<table>
<thead>
<tr>
<th>Original Problem</th>
<th>After preprocessing</th>
<th>Cuts found</th>
<th>#B&amp;B</th>
<th>IP value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns</td>
<td>Rows</td>
<td>Columns</td>
<td>Rows</td>
<td>LP Root</td>
</tr>
<tr>
<td>5198</td>
<td>531</td>
<td>3846</td>
<td>360</td>
<td>30494</td>
</tr>
<tr>
<td>7292</td>
<td>646</td>
<td>5862</td>
<td>488</td>
<td>26977</td>
</tr>
<tr>
<td>8308</td>
<td>801</td>
<td>6235</td>
<td>521</td>
<td>53736</td>
</tr>
<tr>
<td>8627</td>
<td>825</td>
<td>6694</td>
<td>537</td>
<td>49616</td>
</tr>
<tr>
<td>148633</td>
<td>139</td>
<td>138951</td>
<td>139</td>
<td>1181590</td>
</tr>
<tr>
<td>288507</td>
<td>71</td>
<td>202603</td>
<td>71</td>
<td>132878</td>
</tr>
<tr>
<td>1053137</td>
<td>145</td>
<td>370642</td>
<td>90</td>
<td>9950</td>
</tr>
</tbody>
</table>

The following three problems show how much time it takes to get within certain percentages of optimality compared to the total time spent (in seconds).
4.6 References for other results

The above examples illustrate the state-of-the-art on computational results. This section is therefore far from complete. The references given in section 2, contain most of the computational results known by the authors.

5 Alternative techniques

In the last two decades there has been a remarkable development in polyhedral techniques leading to an increase in the size of many combinatorial problems that can be solved by a factor hundred. Most of the computational successes have occurred for zero-one combinatorial problems where the polytope is defined once the dimension is given, such as the traveling salesman problem. For more complex combinatorial optimization problems, and for general integer programming problems less progress has been made. Here we shall give a brief overview of other available solution techniques.

If the number of variables is large compared to the number of constraints column generation may in many cases be a good alternative. It can be viewed as a dual approach to polyhedral techniques in the sense that one aims at generating the extreme points of conv(S) rather than its facets. Instead of solving a separation problem to generate a violated inequality we need to solve the problem of finding a column, i.e. a feasible solution, that can improve the objective function. Column generation was introduced by Gilmore and Gomory (1961) to solve the cutting stock problem. Recent applications are presented by Savelsbergh (1994) and Vanderbeck and Wolsey (1994).

References


E. Balas (1975) "Facets of the knapsack polytope", Mathematical Programming 8 146-164.


E. Balas and A.C. Ho (1980) "Set covering algorithms using cutting planes, heuristics and subgradient optimization" Mathematical Programming Study 12 37-60.

E. Balas and S.M. Ng (1989a) "On the set covering polytope: I. All the facets with coefficients in \{0, 1, 2\}", Mathematical Programming 43 57-69.

E. Balas and S.M. Ng (1989b) "On the set covering polytope: II. Lifting the facets with coefficients in \{0, 1, 2\}", Mathematical Programming 45 1-20.


E.A. Boyd "Fenchel cutting planes for integer programs", Operations Research 42 53-64.

35


M. FISCHETTI (1991b) "Facets of two Steiner arborescence polyhedra", *Mathematical Programming* 51 401–419.


M. GRÖTSCHEL AND O. HOLLAND (1985) "Solving matching problems with linear programming", 37
Mathematical Programming 33 243–259.


38


Overzicht Research Memoranda Rijksuniversiteit Limburg, Faculteit der Economische Wetenschappen vanaf augustus 1994 (tel. 043-883830)

1994

RM/04-026 M.C.M. Vendrik
Dynamics of a Household Norm in Female Labour Supply

RM/04-028 F.H.J. Bunte, M.A. Han, J.A.H. Maks, T.A. Zwick
Quality Competition with Potential Entry: A welfare Analysis of a Quasi-Contestable Market

RM/04-031 M. Galizzi
Wage Changes and Establishment Growth

RM/04-033 M. Filippini, S. Schirm van der Leef
Electricity Demand by Time-of-Use: An Application of Exact Aggregation Theory

RM/04-034 J.A.H. Maks
Competition Policy and Imperfect Information, The Netherlands 1987-1992

Uncertain Technological Progress and Potential Entry

RM/04-036 M. van Wegberg
The Choice between Merger and Alliance in a Collusive Oligopolistic Environment

RM/04-037 V.E van Rijswoud, N.B.J. van der Rijst
The Analysis of Planning Activities using the Demo Transaction Process model

RM/04-038 A. Georges, L. Romme
Changing Business Systems in the Paper and Pulp Industry

RM/04-039 G. Romme

RM/04-040 J.G.A. van Mierlo
De Economische Betekenis voor de Regio van de Spreiding van de Rijksdiensten naar Oostelijk Zuid-Limburg

RM/04-041 R.J.C. Dillen, A.G.L. Romme
Leren door Organisaties

RM/04-042 J.G. Backhaus
The quest for ecological tax reform: A Schumpeterian Approach to Public Finance

RM/04-043 J.G.A. van Mierlo
De Aansluiting tussen Onderwijs en Arbeidsmarkt. Een Bestuurseconomische Verkenning van een Publiek Organisatievraagstuk

RM/04-044 J.G.A. van Mierlo
Wat kunnen Overheden en Ondernemingen van elkaar leren? Een Bestuurseconomisch Perspectief

RM/04-045 F.H.J. Bunte, M.A. Han, J.A.H. Maks, J. Muysken, T.A. Zwick
Product and Process Innovations with Consumer Heterogeneity

RM/04-046 J. Muysken, E. de Regt
A dynamic analysis of structural unemployment in 5 European countries: Austria, Germany, The Netherlands, Sweden and The United Kingdom

1995

RM/95/001 M. Galizzi
Gender Discrimination and Quit Behavior: Do Future Wages Matter? An Analysis for Short-Tenure Workers

RM/95/002 Th. Broecker
Is Cost Sharing for R&D under the Commensurate with Income Standard Actually Arm's Length?

RM/95/003 B. Klaus, H. Peters, J. Storken
Strategy-proof Division with Single-peaked Preferences and Initial Endowments

RM/95/004 J.G. Backhaus
Die Überführung von Produktionsmitteln in Gemeineigentum

RM/95/005 S. Dixon
Limit Pricing and Multi-Market Entry

RM/95/006 M. van Wegberg
Can R&D Alliances Facilitate the Formation of a Cartel?

RM/95/007 A. Persijn, M. Jansen, H. Peters
On The Equivalence of Weakly and Fully Consistent Beliefs in Signaling Games
RM/95/008  C.P.M. van Hoesel, A.P.M. Wagelmans
An $O(T^2)$ algorithm for the economic lot-sizing problem with constant capacities

RM/95/009  M. van Wegberg
Cooperation between Research Companies and Manufacturing Firms: The Choice between Market Contract, Vertical Merger, and an R&D-Alliance

RM/95/010  M. van Wegberg
Mergers and Alliances in the Multimedia Market

RM/95/011  M. van Wegberg
Architectural Battles in the Multimedia Market

RM/95/012  W. Vanhaverbeke
De Economische Uitdagingen voor de Subregio Zuid West-Vlaanderen?

RM/95/013  J. Derks, H. Haller
Weighted Nucleoli

RM/95/014  J.G. Backhaus
The Quest for Ecological Tax Reform: a Schumpeterian Approach to Public Finance

RM/95/015  M.C.M. Vendrik
Dynamics of a household norm in female labour supply, non-linear dynamics

The Working of a Labour Market Model with Potential Entry

RM/95/017  A.J. Vermeulen, M.J.M. Jansen
On the Invariance of Solutions of Finite Games

RM/95/018  W. Vanhaverbeke
Competing with Alliance Networks: the Case of the Risc-Microprocessor Technology

RM/95/019  S. Dixon
Limiting Pricing and the Mode of Entry

RM/95/020  J.G. Backhaus
An Economic Analysis of Constitutional Law

RM/95/021  B. Klaus, H. Peters, T. Storcken
Reallocation of an infinitely divisible good

RM/95/022  S. Van Hoesel, A. Wagelmans, M. Van Vliet
An $O(n \log n)$ algorithm for the two-machine flow shop problem with controllable machine speeds

RM/95/023  K. Aardal, S. Van Hoesel
Polyhedral Techniques in Combinatorial Optimization