A Polyhedral Approach to Single-Machine Scheduling Problems

J.M van den Akker
Department of Mathematics and Computing Science,
Eindhoven University of Technology,
P.O.Box 513, 5600 MB Eindhoven, The Netherlands

C.P.M. van Hoesel
Department of Quantitative Economics,
University of Limburg,
P.O.Box 616, 6200 MD Maastricht, The Netherlands

M.W.P. Savelsbergh
School of Industrial and Systems Engineering,
Georgia Institute of Technology,
Atlanta, GA 30332-0205, USA

Abstract

We report new results for a time-indexed formulation of nonpreemptive single-machine scheduling problems. We give complete characterizations of all facet inducing inequalities with integral coefficients and right-hand side 1 or 2 for the convex hull of the set of feasible partial schedules, i.e., schedules in which not all jobs have to be started. Furthermore, we identify conditions under which these facet inducing inequalities are also facet inducing for the original polytope, which is the convex hull of the set of feasible complete schedules, i.e., schedules in which all jobs have to be started. To obtain insight in the effectiveness of these classes of facet-inducing inequalities, we develop a branch-and-cut algorithm based on them. We evaluate its performance on the strongly $\mathcal{NP}$-hard single machine scheduling problem of minimizing the weighted sum of the completion times subject to release dates.

Key words: scheduling, polyhedral methods, facet inducing inequalities, separation, branch-and-cut.

A Polyhedral Approach to Single-Machine Scheduling Problems

J.M. van den Akker
Department of Mathematics and Computing Science,
Eindhoven University of Technology,
P.O.Box 513, 5600 MB Eindhoven, The Netherlands

C.P.M. van Hoesel
Department of Quantitative Economics,
University of Limburg,
P.O.Box 616, 6200 MD Maastricht, The Netherlands

M.W.P. Savelsbergh
School of Industrial and Systems Engineering,
Georgia Institute of Technology,
Atlanta, GA 30332-0205, USA

Abstract

We report new results for a time-indexed formulation of nonpreemptive single-machine scheduling problems. We give complete characterizations of all facet inducing inequalities with integral coefficients and right-hand side 1 or 2 for the convex hull of the set of feasible partial schedules, i.e., schedules in which not all jobs have to be started. Furthermore, we identify conditions under which these facet inducing inequalities are also facet inducing for the original polytope, which is the convex hull of the set of feasible complete schedules, i.e., schedules in which all jobs have to be started. To obtain insight in the effectiveness of these classes of facet-inducing inequalities, we develop a branch-and-cut algorithm based on them. We evaluate its performance on the strongly $\mathcal{NP}$-hard single machine scheduling problem of minimizing the weighted sum of the completion times subject to release dates.

Key words: scheduling, polyhedral methods, facet inducing inequalities, separation, branch-and-cut.

1 Introduction

Recently developed polyhedral methods have yielded substantial progress in solving many important $\mathcal{NP}$-hard optimization problems. Some well-known examples are the traveling salesman problem [Padberg and Rinaldi 1991], 0-1 integer programming problems [Crowder, Johnson and Padberg 1983], mixed 0-1 integer programming problems [Van

Roy and Wolsey 1987]. We refer to Hoffman and Padberg [1985] and Nemhauser and Wolsey [1988] for general descriptions of the approach.

In comparison, the investigation and development of polyhedral methods for machine scheduling problems is still in its early stages. The majority of the research has focused on single-machine scheduling problems or problems that can be treated as such. Balas [1985] pioneered the study of scheduling polyhedra with his work on the facial structure of the job shop scheduling problem. Queyranne [1993] completely characterized the polyhedron associated with the simple nonpreemptive single-machine scheduling problem. Queyranne and Wang [1991] generalized Queyranne's results to include precedence constraints. Wolsey [1989] compared different formulations for the problem with precedence constraints. Dyer and Wolsey [1990] examined several formulations for the single-machine scheduling problem with release dates, and Nemhauser and Savelsbergh [1992] developed a cutting plane algorithm for this problem. Sousa and Wolsey [1992] investigated a time-indexed formulation for several variants of the nonpreemptive single-machine scheduling problem. Crama and Spieksma [1993] studied the same formulation for problems in which the jobs have equal processing times. Lasserre and Queyranne [1992] presented a mixed integer programming formulation motivated by a control theoretic view of scheduling decisions. We refer to Queyranne and Schulz [1997] for a more comprehensive survey.

In this paper, we report new results for the time-indexed formulation of nonpreemptive single-machine scheduling problems studied by Sousa and Wolsey [1992]. They introduced three classes of inequalities. The first class consists of inequalities with right-hand side 1, and the second and third classes consist of inequalities with right-hand side $k \in \{2, \ldots, n\}$. Furthermore, they developed a cutting plane algorithm based on these three classes of inequalities. They used exact separation algorithms to identify violated inequalities in the first class and violated inequalities with right-hand side 2 in the second class. They used a simple heuristic to identify violated inequalities in the third class. Their computational experiments revealed that the bounds obtained are strong compared to bounds obtained from other mixed integer programming formulations.

These promising computational results stimulated us to study the inequalities with right-hand side 1 or 2 more thoroughly. To do so, we first study the convex hull of the set of feasible partial schedules, i.e., schedules in which not all jobs have to be started. We derive complete characterizations of all facet inducing inequalities with integral coefficients and right-hand side 1 or 2 for this extended polytope. Then, we give conditions under which the identified inequalities are also facet inducing for the original polytope. Our analysis shows that only some of the classes of inequalities used in the computational experiments by Sousa and Wolsey are facet inducing. To obtain insight in the effectiveness of the classes of facet-inducing inequalities we have derived, we have developed a branch-and-cut algorithm based on them. We evaluate its performance on the strongly $\mathcal{NP}$-hard single-machine scheduling problem of minimizing the weighted sum of the completion times subject to release dates.

2 Problem formulation

The usual setting for nonpreemptive single-machine scheduling problems is as follows. A set J of n jobs has to be scheduled on a single machine. Each job $j \in J$ requires uninterrupted processing for a period of length p_j, where p_j is some positive integer. The machine can handle no more than one job at a time.

The time-indexed formulation studied by Sousa and Wolsey [1992] is based on time-discretization, i.e., time is divided into periods, where period t starts at time $t-1$ and ends at time t. The planning horizon is denoted by T, which means that all jobs have to be completed by time T. We assume that $T \geq \sum_{j=1}^{n} p_j$. The formulation is as follows:

$$\text{minimize} \sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt}$$

subject to

$$\sum_{t=1}^{T-p_j+1} x_{jt} = 1 \qquad (j = 1, ..., n), \tag{1}$$

$$\sum_{j=1}^{n} \sum_{s=t-p_j+1}^{t} x_{js} \leq 1 \qquad (t = 1, ..., T), \tag{2}$$

$$x_{jt} \in \{0, 1\} \qquad (j = 1, ..., n;\;\; t = 1, ..., T - p_j + 1),$$

where $x_{jt} = 1$ if job j is started in period t, i.e., at the beginning of period t, and 0 otherwise. This formulation can be used to model several single-machine scheduling problems by an appropriate choice of the objective coefficients and possibly a restriction of the set of variables. For instance, if the objective is to minimize the weighted sum of the start times, we take coefficients $c_{jt} = w_j(t-1)$, where w_j denotes the weight of job j; if there are release dates r_j, i.e., job j becomes available at time r_j, then we discard the variables x_{jt} for $t = 1, \ldots, r_j$. In the sequel, we denote the set of feasible schedules by S.

Many of the single-machine scheduling problems that can be modeled by the time-indexed formulation given above are strongly $\mathcal{NP}$-hard, and therefore the integer programming problem defined by this formulation is strongly $\mathcal{NP}$-hard. Crama and Spieksma [1993] prove that even when we take $p_j = 2$ for all j and $c_{jt} \in \{0, 1\}$ the problem is strongly $\mathcal{NP}$-hard.

In the above formulation, the convex hull P_S of S, the set of feasible schedules, is not full-dimensional. Sousa and Wolsey [1992] showed that, if $T \geq \sum_{j=1}^{n} p_j + p_{\max}$, then $\dim(P_S) = nT - \sum_{j=1}^{n} p_j$, where $p_{\max} = \max\{p_j | j \in \{1, \ldots, n\}\}$. As it is often easier to study full-dimensional polyhedra, we study the convex hull P_{S^*} of S^*, where S^* is the set of all feasible partial schedules, i.e., the set of feasible schedules in which not all jobs have to be started. A description of S^* can be obtained by relaxing the equations (1) into inequalities with sense less-than-or-equal, i.e., the set S^* is described by:

$$\sum_{t=1}^{T-p_j+1} x_{jt} \leq 1 \quad (j = 1, ..., n), \tag{3}$$

$$\sum_{j=1}^{n} \sum_{s=t-p_j+1}^{t} x_{js} \leq 1 \quad (t = 1, ..., T), \tag{4}$$

$$x_{jt} \in \{0, 1\} \qquad (j = 1, ..., n; t = 1, \ldots T - p_j + 1)$$

It is not hard to show that P_{S^*} is full-dimensional. In the sequel, we consider the polytope P_{S^*} unless we state otherwise. When we say that an inequality is valid, we mean that it is valid for P_{S^*}; since P_{S^*} contains P_S, such an inequality is valid for P_S too. Moreover, when we speak about a schedule, we mean a schedule that can be partial, i.e., it does not have to contain all jobs. When the schedule has to contain all jobs we call it a *complete* schedule.

Note that the collection of facet inducing inequalities for the polytope P_{S^*} associated with the set of partial schedules includes all facet inducing inequalities for the polytope P_S associated with the set of complete schedules.

A set $V \subseteq \{0, 1\}^n$ is called *down-monotone* if for all $x, y \in \{0, 1\}^n$ we have that $x \leq y$ and $y \in V$ implies that $x \in V$. Down-monotone 0-1 polytopes are polytopes that are the convex hull of a down-monotone subset of $\{0, 1\}^n$. Hammer, Johnson, and Peled [1975] studied monotone polytopes and proved the following lemma.

Lemma 1 [Hammer, Johnson, and Peled, 1975]
Let P be a down-monotone 0-1 polytope. A facet inducing inequality $ax \leq b$ for P with integral coefficients a_j and integral right-hand side b has either $b > 0$ and coefficients a_j in $\{0, 1, \ldots, b\}$ or is a positive scalar multiple of $-x_j \leq 0$ for some j. □

Since P_{S^*} is a down-monotone 0-1 polytope, the result holds for P_{S^*} too. The above lemma implies that all facet inducing inequalities with right-hand side 0 for P_{S^*} have the form $x_{js} \geq 0$. It can be shown that each inequality $x_{js} \geq 0$ is facet inducing for P_{S^*} by observing that all other unit vectors together with the all-zero vector are affinely independent. Extending the proof of Crama and Spieksma [1993], we can show that these inequalities are also facet inducing for P_S, if $T \geq \sum_{j=1}^{n} p_j + p_{\max}$.

Before we present our analysis of the structure of facet inducing inequalities with right-hand side 1 or 2, we introduce some notation and definitions.

The index-set of variables with nonzero coefficients in an inequality is denoted by V. The set of variables with nonzero coefficients in an inequality associated with job j defines a set of time periods $V_j = \{s|(j,s) \in V\}$. If job j is started in period $s \in V_j$, then we say that job j is started in V. With each set V_j we associate two values

$$l_j = \min\{s|s - p_j + 1 \in V_j\}$$

and

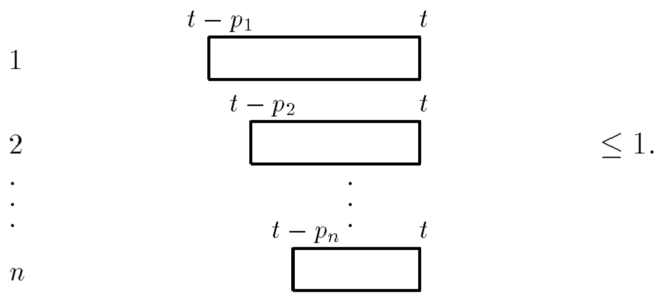
$$u_j = \max\{s|s \in V_j\}.$$

For convenience, let $l_j = \infty$ and $u_j = -\infty$ if $V_j = \emptyset$. Note that if $V_j \neq \emptyset$, then l_j is the first period in which job j can be finished if it is started in V, and that u_j is the last period in which job j can be started in V. Furthermore, let $l = \min\{l_j|j \in \{1,...,n\}\}$ and $u = \max\{u_j|j \in \{1,...,n\}\}$.

We define an interval $[t_1, t_2]$ as the set of periods $\{t_1+1, t_1+2, \ldots, t_2\}$, i.e., the set of periods between time t_1 and time t_2. If $t_1 \geq t_2$, then $[t_1, t_2] = \emptyset$.

For presentational convenience, we use $x(S)$ to denote $\sum_{(j,s)\in S} x_{js}$. Recall that P_{S^*} is a down-monotone 0-1 polytope. As a consequence of Lemma 1, valid inequalities with right-hand side 1 will be denoted by $x(V) \leq 1$ and valid inequalities with right-hand side 2 will be denoted by $x(V^1) + 2x(V^2) \leq 2$, where $V = V^1 \cup V^2$ and $V^1 \cap V^2 = \emptyset$. Furthermore, we define $V_j^2 = \{s \mid (j,s) \in V^2\}$.

In the sequel, we shall often represent inequalities by diagrams. A diagram contains a line for each job. The blocks on the line associated with job j indicate the time periods s for which x_{js} occurs in the inequality. For example, an inequality of the form (4) can be represented by the following diagram:



3 Facet inducing inequalities with right-hand side 1

The purpose of this section is twofold. First, we present new results that extend and complement the work of Sousa and Wolsey [1992]. Second, we familiarize the reader with our approach in deriving complete characterizations of classes of facet inducing inequalities. Note that parts of the analysis in this section can be simplified, but we present it in this way to demonstrate the approach we use to deal with the more complicated case of right-hand side 2.

Establishing complete characterizations of facet inducing inequalities with right-hand side 1 for the extended polytope P_{S^*} proceeds in three phases. First, we derive necessary conditions in the form of various structural properties of facet-inducing inequalities. These properties follow from the observation that a valid inequality $x(V) \leq 1$ can be facet inducing only if it is maximal, i.e., if there does not exist a valid inequality $x(W) \leq 1$ with $V \subset W$ where V is a proper subset of W. Second, once we have these structural properties, we derive classes of inequalities that contain all facet inducing inequalities. Finally, we show that the maximality is also sufficient.

Then we show that under mild conditions on the horizon T we can guarantee that the facet inducing inequalities we derived for P_{S^*} are also facet inducing for the original polytope P_S.

Recall that we do not require a schedule to contain all jobs.

Lemma 2 *A facet inducing inequality $x(V) \leq 1$ for P_{S^*} is maximal.* □

Property 1 *If $x(V) \leq 1$ is valid and maximal, then the sets V_j are intervals, i.e., $V_j = [l_j - p_j, u_j]$, for $j = 1, \ldots, n$.*

Proof. Let $j \in \{1, \ldots, n\}$ and assume $V_j \neq \emptyset$. By definition $l_j - p_j + 1$ is the smallest s such that $s \in V_j$ and u_j is the largest such value. Consider any s with $l_j - p_j + 1 < s < u_j$ and let job j be started in period s, i.e., $x_{js} = 1$.

Suppose $(i, t) \in V$ is such that $x_{it} = x_{js} = 1$ defines a feasible schedule. If $t < s$, i.e., job i is started before job j, then the schedule that we obtain by postponing the start of job j until period u_j is also feasible. This schedule does not satisfy $x(V) \leq 1$, which contradicts the validity of the inequality. Hence no job can be started in V before job j. Similarly, we obtain a contradiction if $t > s$, which implies that no job can be started in V after job j.

We conclude that choosing $x_{js} = 1$ prohibits any job from starting in V. Because of the maximality of $x(V) \leq 1$, we must have $(j, s) \in V$. □

Property 2 *Let $x(V) \leq 1$ be valid and maximal.*
(a) *Assume $l = l_1 \leq l_2 = \min\{l_j | j \in \{2, ..., n\}\}$. Then $V_1 = [l - p_1, l_2]$ and $V_j = [l_j - p_j, l]$ for all $j \in \{2, ..., n\}$.*
(b) *Assume $u = u_1 \geq u_2 = \max\{u_j | j \in \{2, ..., n\}\}$. Then $V_1 = [u_2 - p_1, u]$ and $V_j = [u - p_j, u_j]$ for all $j \in \{2, ..., n\}$.*

Proof. (a) Suppose that $l = l_1 \leq l_2 = \min\{l_j \mid j \in \{2, \ldots, n\}\}$. Observe that Property 1 implies that V_1 is an interval and that by definition its lower bound equals $l - p_1$. We now show that the upper bound is equal to l_2. Since $x_{2,l_2-p_2+1} = 1$ and $x_{1s} = 1$ defines a feasible schedule for any $s > l_2$, we have that only one of these variables can occur in $x(V) \leq 1$; as by definition $(2, l_2 - p_2 + 1) \in V$, it follows that the upper bound of V_1 is at most l_2. Now, let $x_{1s} = 1$ for some $s \in [l - p_1, l_2]$. Reasoning as in the proof of Property 1 we can show that since $l - p_1 + 1 \in V_1$ it follows that no job can be started in V after job 1. As $s \leq l_2 = \min\{l_j | j \in \{2, ..., n\}\}$, it is impossible to start any job in V before job 1. From the maximality of $x(V) \leq 1$ we conclude that $V_1 = [l - p_1, l_2]$. Similar arguments can be applied to show that $V_j = [l_j - p_j, l]$ for all $j \in \{2, \ldots, n\}$.

The proof of (b) is similar to that of (a). □

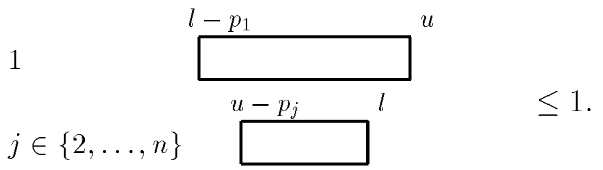
Observe that by Property 2(a) a valid and maximal inequality $x(V) \leq 1$ with $l = l_1$ necessarily has $u_1 = u$. Consequently, Lemma 2 and Properties 2(a) and 2(b) can be combined to give the following theorem.

Theorem 1 *A facet inducing inequality $x(V) \leq 1$ for P_{S^*} has the following structure:*

$$\begin{aligned} &V_1 = [l - p_1, u], \\ &V_j = [u - p_j, l] \quad (j \in \{2, ..., n\}), \end{aligned} \tag{5}$$

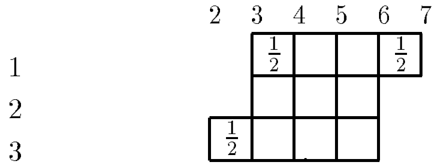
where $l = l_1 \leq u_1 = u$. □

Theorem 1 says that a facet inducing inequality with right-hand side 1 can be represented by the following diagram:



Note that if $l = u$, then the inequalities with structure (5) coincide with the inequalities (4); if $l = p_1$, $u = T - p_1 + 1$, and $V_j = \emptyset$ for all $j \in \{2, \ldots, n\}$, then the inequalities with structure (5) coincide with the inequalities (3).

Example 1 Let $n = 3$, $p_1 = 3$, $p_2 = 4$ and $p_3 = 5$. The inequality with structure (5), $l = l_1 = 6$ and $u = u_1 = 7$ is given by the following diagram:



$\leq 1.$

Note that the fractional solution $x_{14} = x_{17} = x_{33} = \frac{1}{2}$ satisfies (3) and (4), but violates the above inequality.

The following theorem shows that the given necessary conditions are also sufficient. The proof of this theorem uses the concept of a *counterexample*. If $x(V) \leq 1$ is maximal, then for each $(j, s) \notin V$, there exists a $(j', s') \in V$ such that $x_{js} = x_{j's'} = 1$ is a feasible schedule, since the variable x_{js} could be added to the inequality otherwise. Such a schedule is called a *counterexample* for (j, s).

Theorem 2 *A valid inequality $x(V) \leq 1$ is facet inducing for P_{S^*} if and only if it is maximal.*

Proof. Lemma 2 already states that a facet inducing inequality $x(V) \leq 1$ for P_{S^*} is maximal. Now, let $x(V) \leq 1$ be valid and maximal. Let $F = \{x \in P_{S^*} | x(V) = 1\}$. We show $\dim(F) = \dim(P_{S^*}) - 1$ by exhibiting $\sum_{j=1}^{n} T - p_j + 1$ affinely independent vectors in F. First, take all unit vectors $x_{js} = 1$ with $(j, s) \in V$. Then, because of the maximality of $x(V) \leq 1$, there exists a counterexample for each $(j, s) \notin V$. Together with the unit vectors, these counterexamples provide the set of affinely independent vectors. □

Corollary 1 *A valid inequality $x(V) \leq 1$ with structure* (5) *that is maximal is facet inducing for P_{S^*}.* □

Sousa and Wolsey already established that the class of inequalities with structure (5) is facet inducing for P_S if the horizon T is large enough.

Theorem 3 [Sousa and Wolsey, 1992]
If $T \geq \sum_{j=1}^{n} p_j + 3p_{\max}$, then a valid inequality $x(V) \leq 1$ with structure (5) *that is maximal is facet inducing for P_S, where $p_{\max} = \max\{p_j | j = 1, \ldots, n\}$.* □

A valid inequality $x(V) \leq 1$ with structure (5) is maximal if and only if the following conditions hold. First, we need that either $V_j \neq \emptyset$ for some $j \in \{2, \ldots, n\}$, or $l = p_1$ and $u = T - p_1 + 1$; these conditions ensure that an inequality containing only one job includes all variables associated with this job. Second, we need conditions to exclude nonmaximal inequalities at the border of the interval $[0, T]$. If $l = u$, then we must have

$p_{[2]} \leq l \leq T - p_{[2]} + 1$, where $p_{[2]}$ denotes the processing time of the smallest job but one. If $l < u$, then we must have $l \geq p_1$ and $u \geq \min\{p_j | j \neq 1, p_j > u - l\}$ and $u \leq T - p_1 + 1$ and $l \leq T + 1 - \min\{p_j | j \neq 1, p_j > u - l\}$.

Recall that the inequalities (4) coincide with inequalities with structure (5) with $l = u$ and that the inequalities (3) coincide with inequalities with structure (5) with $l = p_1$, $u = T - p_1 + 1$, and $V_j = \emptyset$ for all $j \in \{2, \ldots, n\}$. The above maximality conditions imply that inequalities (4) are maximal for $p_{[2]} \leq t \leq T - p_{[2]} + 1$ and that inequalities (3) are maximal for each job j_1 with $T \geq 2p_{j_1} + \max\{p_j | j \neq j_1\}$. Hence, if T is large enough, then inequalities (3) are all maximal.

Note that an inequality with structure (5) is determined by one job, which without loss of generality is called job 1, and two time periods l and u. Since the maximality condition, stating that $V_j \neq \emptyset$ for some $j \in \{2, \ldots, n\}$, implies that $u - p_{\max} \leq l$, it follows that the number of facet inducing inequalities with structure (5) that does not coincide with an inequality (3) is bounded by $nTp_{\max}$, and hence the total number of facet inducing inequalities with structure (5) is bounded by $nTp_{\max} + n$, which is polynomial in the size of the formulation.

4 Facet inducing inequalities with right-hand side 2

In the previous section, we have derived a complete characterization of all facet inducing inequalities with right-hand side 1 for the extended polytope P_{S^*}. Through a similar analysis, we now derive a characterization of all facet inducing inequalities with right-hand side 2 for P_{S^*}. First, we consider the structure of valid inequalities $x(V^1) + 2x(V^2) \leq 2$ carefully. We find that in such an inequality we can distinguish three sets of variables, which we will call L, M, and U; consequently, we call the corresponding structure the *LMU-structure*. Then, based on this LMU-structure, we derive a characterization of facet inducing inequalities with right-hand side 2 for P_{S^*}. Finally, we give mild conditions on the horizon T under which the identified inequalities are facet inducing for the original polytope P_S too. Recall that a schedule does not have to contain all jobs.

We start by studying the structure of valid inequalities with right-hand side 2 and coefficients 0, 1, and 2. Consider a valid inequality $x(V^1) + 2x(V^2) \leq 2$. Clearly, at most two jobs can be started in $V = V^1 \cup V^2$. Let $j \in \{1, ..., n\}$ and $s \in V_j$. It is easy to see that, if job j is started in period s, at least one of the following three statements is true.

(i) It is impossible to start any job in V before job j, and at most one job can be started in V after job j.

(ii) There exists a job i with $i \neq j$ such that job i can be started in V before as well as after job j and any job j' with $j' \neq j, i$ cannot be started in V.

(iii) At most one job can be started in V before job j, and it is impossible to start any job in V after job j.

Therefore, we can write $V = L \cup M \cup U$, where $L \subseteq V$ is the set of variables for which statement (i) holds, $M \subseteq V$ is the set of variables for which statement (ii) holds, and $U \subseteq V$ is the set of variables for which statement (iii) holds. Analogously, we can write $V_j = L_j \cup M_j \cup U_j$. Note that each of the sets L_j, M_j, and U_j may be empty.

If job j is started in a period in V_j^2, then it is impossible to start any job in V before or after job j. It follows that $V_j^2 \subseteq L_j \cap U_j$ for all j and hence $V^2 \subseteq L \cap U$. It is not hard to see that, if $L_j \neq \emptyset$ and $U_j \neq \emptyset$, then the minimum element in L_j is less than or equal to the minimum element in U_j, and the maximum element in L_j is less than or equal to the maximum element in U_j. By definition $L_j \cap M_j = \emptyset$ and $M_j \cap U_j = \emptyset$. The set M_j consists of periods between the maximum element of L_j and the minimum element of U_j and hence M_j must be empty if $L_j \cap U_j \neq \emptyset$. By definition of the sets L and U, $x(L) \leq 1$ and $x(U) \leq 1$ are valid inequalities.

We conclude that a valid inequality $x(V^1) + 2x(V^2) \leq 2$ can be represented by a collection of sets L_j, M_j, and U_j. To derive necessary conditions on the structure of facet inducing inequalities with right-hand side 2, we study this LMU-structure more closely.

In the case of right-hand side 1, we only needed the concept of maximality to derive the structural properties. In this case, we also need the concept of nondecomposability. A valid inequality $x(V^1) + 2x(V^2) \leq 2$ is called *nondecomposable* if it cannot be written as the sum of two valid inequalities $x(W) \leq 1$ and $x(W') \leq 1$. The concept of maximality becomes a little more complex in this case. A valid inequality $x(V^1) + 2x(V^2) \leq 2$ is called *maximal* if there does not exist a valid inequality $x(W^1) + 2x(W^2) \leq 2$ with $V \subseteq W$, $V^2 \subseteq W^2$, where at least one of the subsets is a proper subset. The following lemma yields a general necessary condition and will be frequently used to prove structural properties.

Lemma 3 *A facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} is nondecomposable and maximal.* $\Box$

The remaining part of the analysis of the LMU-structure proceeds in two phases. In the first phase, we derive conditions on the structure of the sets L and U by considering each of them separately. After having characterized the structure of L and U, it turns out that, when considering the overall LMU-structure, we have to distinguish three situations, one for each possible way of combining L and U. In the second phase, we characterize the set M for each of these three situations.

Property 3 *If $x(V^1) + 2x(V^2) \leq 2$ is valid and maximal, then the sets L_j, M_j, and U_j $(j = 1, \ldots, n)$ are intervals. Moreover, $V_j^2 = L_j \cap U_j$ for all j, i.e., $V^2 = L \cap U$.*

Proof. The proof of the first part is similar to the proof of Property 1; the second part is trivial. □

Property 4 *Let $x(V^1) + 2x(V^2) \leq 2$ be valid, nondecomposable, and maximal.*
(a) *Assume $l = l_1 \leq l_2 \leq \min\{l_j \mid j \in \{3, \ldots, n\}\}$. Then $L_1 = [l - p_1, l_2]$ and $L_j = [l_j - p_j, l]$ for all $j \in \{2, \ldots, n\}$. Furthermore, there exists a $j \in \{2, \ldots, n\}$ such that $L_j \neq \emptyset$.*
(b) *Assume $u = u_1 \geq u_2 \geq \max\{u_j \mid j \in \{3, \ldots, n\}\}$. Then $U_1 = [u_2 - p_1, u]$ and $U_j = [u - p_j, u_j]$ for all $j \in \{2, \ldots, n\}$. Furthermore, there exists a $j \in \{2, \ldots, n\}$ such that $U_j \neq \emptyset$.*

Proof. (a) The proof for $L_1 = [l - p_j, l_2]$ and $L_j = [l_j - p_j, l]$ for all $j \in \{2, \ldots, n\}$ is similar to the proof of Property 2. Now suppose that $L_j = \emptyset$ for all $j \in \{2, \ldots, n\}$. Then $x(V^1) + 2x(V^2) \leq 2$ can be written as the sum of the valid inequalities $x(W) \leq 1$ and $x(W') \leq 1$, where $W = \{(1, s) \mid s \in L_1 \cap U_1\} \cup \{(j, s) \mid j \in \{2, \ldots, n\}, s \in V_j\}$ and $W' = \{(1, s) \mid s \in V_1\}$. This contradicts the fact that $x(V^1) + 2x(V^2) \leq 2$ is nondecomposable.

The proof of (b) is similar to that of (a). □

Like the proof of Theorem 2, many of the proofs of the properties and theorems presented in this section use the concept of a counterexample. If $x(V^1) + 2x(V^2) \leq 2$ is maximal, then for any $(j, s) \notin V$ there must exist a feasible schedule such that $x_{js} = 1$ and $x(V^1) + 2x(V^2) = 2$; this schedule is called a *counterexample* for (j, s).

The following property plays a crucial role in the characterization. It shows that a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ has at most three types of interval L_j and at most three types of interval U_j.

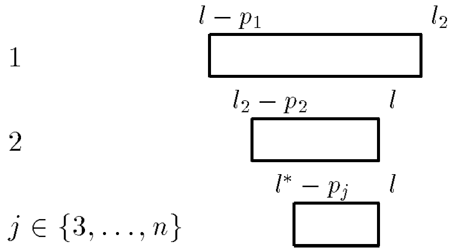
Property 5 *Let $x(V^1) + 2x(V^2) \leq 2$ be valid, nondecomposable, and maximal.*
(a) *Assume $l = l_1 \leq l_2 \leq l^*$, where $l^* = \min\{l_j \mid j \in \{3, \ldots, n\}\}$. Then for all $j \in \{3, \ldots, n\}$ with $L_j \neq \emptyset$ we have $l_j = l^*$ and for all $j \in \{3, \ldots, n\}$ with $L_j = \emptyset$ we have $l^* - p_j \geq l$, i.e., $L_j = [l^* - p_j, l]$ for all $j \in \{3, \ldots, n\}$.*

(b) *Assume* $u = u_1 \geq u_2 \geq u^*$, *where* $u^* = \max\{u_j \mid j \in \{3, \ldots, n\}\}$. *Then for all* $j \in \{3, \ldots, n\}$ *with* $U_j \neq \emptyset$ *we have* $u_j = u^*$ *and for all* $j \in \{3, \ldots, n\}$ *with* $U_j = \emptyset$ *we have* $u^* \leq u - p_j$, *i.e.,* $U_j = [u - p_j, u^*]$ *for all* $j \in \{3, \ldots, n\}$.

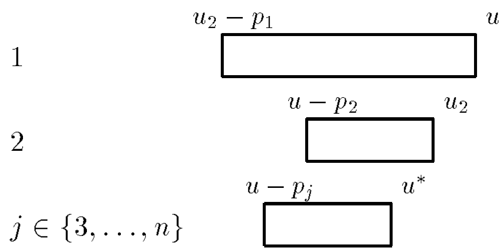
Proof. (a) Let $x(V^1) + 2x(V^2) \leq 2$ be valid and maximal with $l = l_1 \leq l_2 \leq l^*$. By definition of l^* and Property 4, $L_j \subseteq [l^* - p_j, l]$ for all $j \in \{3, \ldots, n\}$. We assume without loss of generality $l^* = l_3$. Suppose that $L_j \neq [l^* - p_j, l]$ for some $j \in \{4, \ldots, n\}$, say $L_4 \neq [l^* - p_4, l]$. Clearly, if $l^* - p_4 \geq l$, then $L_4 = \emptyset = [l^* - p_4, l]$. Consequently, $l^* - p_4 < l$ and $l_4 > l^*$, i.e., $l^* - p_4 + 1 \notin V_4$. Since $x(V^1) + 2x(V^2) \leq 2$ is maximal, there exists a counterexample for $(4, l^* - p_4 + 1)$. Let $x_{4,l^*-p_4+1} = x_{j_1 s_1} = x_{j_2 s_2} = 1$ define such a counterexample. Since $l^* - p_4 + 1 \leq l$, the jobs j_1 and j_2 are started after job 4. Clearly one of the jobs $1, 2$ and 3 does not occur in $\{j_1, j_2\}$. Suppose job 3 does not occur. It is now easy to see that $x_{3,l^*-p_3+1} = x_{j_1 s_1} = x_{j_2 s_2} = 1$ is a feasible schedule, which contradicts the validity of $x(V^1) + 2x(V^2) \leq 2$. If job 1 or job 2 does not occur in $\{j_1, j_2\}$ we obtain a contradiction in the same way.

The proof of (b) is similar to that of (a). □

The combination of Lemma 3 and Properties 4 and 5 shows that, if $x(V^1) + 2x(V^2) \leq 2$ is facet inducing for P_{S^*} and if $l = l_1 \leq l_2 \leq l^*$, then the set L can be represented by the following diagram:



Similarly, if $u = u_1 \geq u_2 \geq u^*$, then the set U can be represented by the following diagram:



The diagrams clearly show that a facet inducing inequality with right-hand side 2 for P_{S^*} contains at most three types of intervals L_j and at most three types of intervals U_j. The intervals L_j are characterized by the definition of the first period of the interval; the intervals U_j are characterized by the definition of the last period of the interval. In fact, the intervals L_j have the same structure for all but two jobs; the same holds for the intervals U_j.

It turns out that, when we study the overall LMU-structure, it suffices to consider three situations, based on the jobs with the deviant intervals L_j and U_j:

(1a) $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$, where $l^* = \min\{l_j \mid j \in \{3, \ldots, n\}\}$ and $u^* = \max\{u_j \mid j \in \{3, \ldots, n\}\}$;

(1b) $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$, where $l^* = \min\{l_j \mid j \in \{3, \ldots, n\}\}$ and $u^* = \max\{u_j \mid j \in \{2, 4, \ldots, n\}\}$;

(2) $l = l_1$ and $u = u_2$.

Before we investigate each of the three situations, we present a property that applies to Case 1.

Property 6 *If $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1 < l_2 = \min\{l_j \mid j \in \{2, \ldots, n\}\}$ and $u = u_1 > u_i = \max\{u_j \mid j \in \{2, \ldots, n\}\}$ is valid, nondecomposable, and maximal, then $l_2 < u_i$.*

Proof. The proof is based on the fact that if $l_2 \geq u_i$, then $x(V^1) + 2x(V^2) \leq 2$ can be written as the sum of two valid inequalities with right-hand side 1. □

4.1 Case (1a)

The conditions on l_j and u_j and Properties 4 and 5 completely determine the sets L and U. Therefore, all that remains to be investigated is the structure of the set M.

Property 7 *If $x(V^1) + 2x(V^2) \leq 2$ is valid, nondecomposable, and maximal with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$, then*

$$\begin{aligned}
M_1 &= [u^* - p_1, l^*] \cap [l_2, u_2 - p_1], \\
M_2 &= [u^* - p_2, l^*] \cap [l, u - p_2] \cap [l_2 - p_2, u_2], \\
M_j &= [u_2 - p_j, l_2] \cap [l, u - p_j], \qquad (j \in \{3, \ldots, n\}).
\end{aligned}$$

Proof. Let $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ be valid, nondecomposable, and maximal. We derive the structure of the set M from that of L and U.

Each set M_j is the intersection of three intervals. The first interval follows from the condition that there exists a job i with $i \neq j$ such that if job j is started in M_j, then job i can be started in V before as well as after job j. The second interval follows from the condition that if job j is started in M_j, then any job j' with $j' \neq j, i$ cannot be started in V. The third interval is $[l_j - p_j, u_j]$.

We first determine M_1. If job 1 is started in M_1, then, since $l_2 \leq l^*$ and $u_2 \geq u^*$, job 2 is the job that can be started in V before as well as after job 1. This implies that $M_1 \subseteq [l_2, u_2 - p_1]$. Furthermore, it is impossible to start any job $j \in \{3, \ldots, n\}$ in V and hence $M_1 \subseteq [u^* - p_1, l^*]$. We conclude that $M_1 \subseteq [u^* - p_1, l^*] \cap [l_2, u_2 - p_1]$. Clearly, this dominates the condition that $M_1 \subseteq [l - p_1, u]$. If job 1 is started in period $s \in [u^* - p_1, l^*] \cap [l_2, u_2 - p_1]$, then, since $s \in [l_2, u_2 - p_1]$, $[l_2, u_2 - p_1] \subset [l - p_1, u]$, and $L_2 \cap U_2 = [u - p_2, l]$, job 2 cannot be started in $L_2 \cap U_2$. Since $x(V^1) + 2x(V^2) \leq 2$ is maximal, it follows that $M_1 = [u^* - p_1, l^*] \cap [l_2, u_2 - p_1]$.

The other sets M_j can be determined in the same way. The condition $M_j \subseteq [l_j - p_j, u_j]$ is dominated by other conditions for all but $j = 2$. □

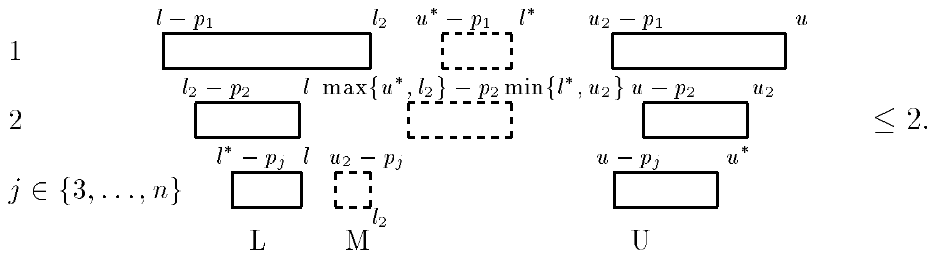
Lemma 3 and Properties 4, 5 and 7 completely determine the LMU-structure of a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$. We can further show that for all $j \in \{3, \ldots, n\}$ we have $[u_2 - p_j, l] \subseteq L_j$ and $[u - p_j, l_2] \subseteq U_j$. We combine all these results to obtain the following theorem. Note that we have reformulated the intervals M_j to emphasize their inherent structure.

Theorem 4 *A facet inducing inequality $x(V^1)+2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ has the following LMU-structure:*

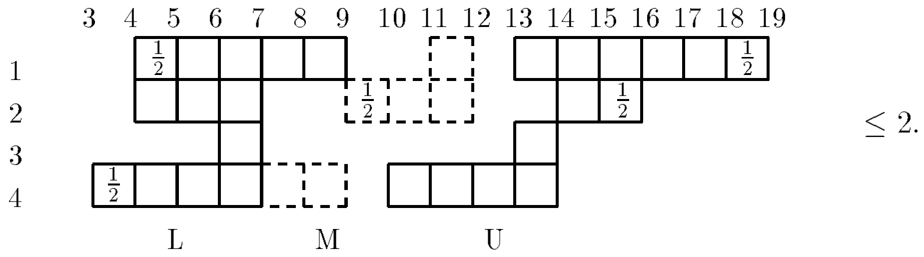
$$\begin{aligned}
&L_1 = [l - p_1, l_2], \quad M_1 = [u^* - p_1, l^*] \setminus (L_1 \cup U_1),\\
&L_2 = [l_2 - p_2, l], \quad M_2 = [\max\{u^*, l_2\} - p_2, \min\{l^*, u_2\}] \setminus (L_2 \cup U_2),\\
&L_j = [l^* - p_j, l], \quad M_j = [u_2 - p_j, l_2] \setminus (L_j \cup U_j),\\
&\\
&U_1 = [u_2 - p_1, u],\\
&U_2 = [u - p_2, u_2],\\
&U_j = [u - p_j, u^*] \quad (j \in \{3, \ldots, n\}),
\end{aligned} \tag{6}$$

where $[u_2 - p_j, l] \subseteq L_j$ and $[u - p_j, l_2] \subseteq U_j$ for all $j \in \{3, \ldots, n\}$. □

Hence, a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ can be represented by the following diagram:



Example 2 Let $n = 4$, $p_1 = 3$, $p_2 = 5$, $p_3 = 6$, and $p_4 = 9$. The inequality with LMU-structure (6) and $l = l_1 = 7$, $l_2 = 9$, $l^* = 12$, $u^* = 14$, $u_2 = 16$ and $u = u_1 = 19$ is given by the following diagram:



Note that the fractional solution $x_{15} = x_{1,19} = x_{2,10} = x_{2,16} = x_{4,4} = \frac{1}{2}$ violates this inequality. It is easy to check that this solution satisfies all inequalities with structure (5).

The following theorem shows that the given necessary conditions are also sufficient.

Theorem 5 *A valid inequality $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ and LMU-structure* (6) *that is nondecomposable and maximal is facet inducing for P_{S^*}.*

Proof. Let $x(V^1) + 2x(V^2) \leq 2$ be a valid inequality with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ and LMU-structure (6) that is nondecomposable and maximal, and let $F = \{x \in P_{S^*} | x(V^1) + 2x(V^2) = 2\}$. We show that $\dim(F) = \dim(P_{S^*}) - 1$ by exhibiting $\dim(P_{S^*}) - 1$ linearly independent directions in F, where a direction is a vector $d = x - y$ with $x, y \in F$. For notational convenience, a direction will be specified by its nonzero components. We give three sets of directions: unit vectors $d_{js} = 1$ for all $(j, s) \notin V$, vectors $d_{js} = 1, d_{1,l-p_1+1} = d_{2u_2} = -1$ for all $(j, s) \in V^2$, and a set of $|V| - |V^2| - 1$ linearly independent directions $d_{j_1 s_1} = 1, d_{j_2,s_2} = -1$ with $(j_1, s_1), (j_2, s_2) \in V \setminus V^2$. Together these give $\dim(P_{S^*}) - 1$ linearly independent directions in F.

If $(j,s) \notin V$, then, since $x(V^1) + 2x(V^2) \leq 2$ is maximal, there is a counterexample for (j,s), say, defined by $x_{js} = x_{j_1 s_1} = x_{j_2 s_2} = 1$. Clearly this schedule is an element of F. Note that the schedule $y_{j_1 s_1} = y_{j_2 s_2} = 1$ also is an element of F and hence $d = x - y$ yields the direction $d_{js} = 1$.

For $(j,s) \in V^2$ the schedule defined by $x_{js} = 1$ is an element of F. Since $l < l_2$ and, by Property 6, $l_2 < u_2$, we have that $y_{1,l-p_1+1} = y_{2u_2} = 1$ defines a feasible schedule. This schedule is also an element of F and hence $d_{js} = 1, d_{1,l-p_1+1} = d_{2u_2} = -1$ is a direction in F for all $(j,s) \in V^2$.

The remaining $|V| - |V^2| - 1$ directions have the form $d_{j_1 s_1} = 1, d_{j_2 s_2} = -1$ with $(j_1, s_1), (j_2, s_2) \in V \setminus V^2$. We determine the directions in such a way that the undirected graph G with vertex set $V \setminus V^2$ and with edge set equal to the pairs $(j_1, s_1), (j_2, s_2)$ corresponding to the chosen directions forms a spanning tree. This guarantees that the determined directions are linearly independent. We refer to Van den Akker [1994] for a complete description of the determination of these directions. □

The following theorem shows that the sufficient conditions given by the previous theorem are also sufficient for the original polytope if the planning horizon T is large enough.

Theorem 6 *If $T \geq \sum_{j=1}^{n} p_j + 5p_{\max}$, then a valid inequality $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_2 \geq u^*$ and LMU-structure* (6) *that is nondecomposable and maximal is facet inducing for P_S.*

Proof. The proof proceeds along the same lines as the proof of the previous theorem: we pick almost the same set of directions. We need the extra term of $5p_{\max}$ in the bound on T, because we now have to extend the partial schedules used to define the directions to complete schedules. □

An inequality with LMU-structure (6) is determined by two jobs and six time periods l, l_2, l^*, u^*, u_2 and u. It is facet inducing for P_{S^*} if and only if it is maximal and nondecomposable. We have derived the exact conditions on the eight defining parameters to ensure this. These conditions are omitted here for reasons of brevity. We refer to Van den Akker [1994] for a complete description. It turns out that the number of facet inducing inequalities for P_{S^*} with structure (6) is bounded by $2n^2T^3p_{\max}^3$, and is hence polynomial in the size of the formulation.

4.2 Case (1b)

Like in Case (1a), the conditions on l_j and u_j and Properties 4 and 5 completely determine the sets L and U. From these properties we derive that, if $l_2 = l^*$ and $u_3 = u^*$, then $L_i \neq \emptyset$ and $U_i \neq \emptyset$, where i is such that $p_i = \max\{p_j \mid j \in \{2, \ldots, n\}\}$. But then

$l_i = l_2$ and $u_i = u_3$ and we are in Case (1a). We conclude that $l_2 < l^*$ or $u_3 > u^*$. All that remains to be investigated is the structure of the set M.

Property 8 *If $x(V^1) + 2x(V^2) \leq 2$ is valid, nondecomposable, and maximal with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$, then*

$$\begin{aligned}
&M_1 = \emptyset, \\
&M_2 = [u_3 - p_2, l^*] \cap [l, u - p_2] \cap [l_2 - p_2, u^*], \\
&M_3 = [u^* - p_3, l_2] \cap [l, u - p_3] \cap [l^* - p_3, u_3], \\
&M_j = [u_3 - p_j, l_2] \cap [l, u - p_j] \qquad\qquad j \in \{4, \ldots, n\}.
\end{aligned}$$

Proof. The proof is analogous to the proof of Property 7. □

Properties 4, 5, and 8 determine the LMU-structure of a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$. Like in Case (1a), we use a different representation of the set M to emphasize the inherent structure of the intervals M_j. It turns out that a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j < l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$ has the following property, which restricts the class of inequalities determined by Properties 4, 5, and 8 and leads to a simpler form of the intervals M_j.

Property 9 *If $x(V^1) + 2x(V^2) \leq 2$ is valid, nondecomposable, and maximal with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$, then $l^* \leq u^*$.* □

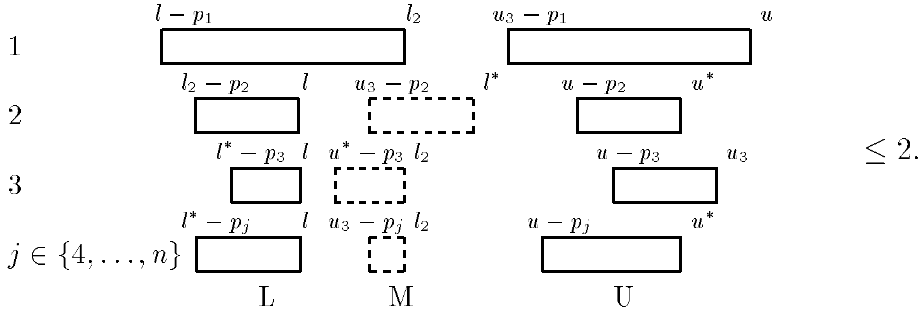
Lemma 3 and Properties 4, 5, 8, and 9 can be combined to give the following theorem.

Theorem 7 *A facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$ has the following LMU-structure:*

$$\begin{array}{llll}
L_1 = [l - p_1, l_2], & M_1 = \emptyset, & U_1 = [u_3 - p_1, u], & \\
L_2 = [l_2 - p_2, l], & M_2 = [u_3 - p_2, l^*] \setminus (L_2 \cup U_2), & U_2 = [u - p_2, u^*], & \\
L_3 = [l^* - p_3, l], & M_3 = [u^* - p_3, l_2] \setminus (L_3 \cup U_3), & U_3 = [u - p_3, u_3], & \\
L_j = [l^* - p_j, l], & M_j = [u_3 - p_j, l_2] \setminus (L_j \cup U_j), & U_j = [u - p_j, u^*] & (j \in \{4, \ldots, n\}),
\end{array} \tag{7}$$

where $l^* \leq u^*$. □

Hence, a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, and $l_j > l_2$ or $u_j < u_3$ for all $j \in \{2, \ldots, n\}$ can be represented by the following diagram:



The following theorem shows that the given necessary conditions are also sufficient.

Theorem 8 *A valid inequality* $x(V^1) + 2x(V^2) \leq 2$ *with* $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, *and* $l_j > l_2$ *or* $u_j < u_3$ *for all* $j \in \{2, \ldots, n\}$ *and LMU-structure* (7) *that is nondecomposable and maximal is facet inducing for* P_{S^*}.

Proof. The proof of this theorem is similar to that of Theorem 5. □

In the same way as in Case (1a), the proof can be extended to prove that the sufficient conditions given by the previous theorem are also sufficient for the original polytope if the horizon T is large enough.

Theorem 9 *If* $T \geq \sum_{j=1}^{n} p_j + 5p_{\max}$, *then a valid inequality* $x(V^1) + 2x(V^2) \leq 2$ *with* $l = l_1 < l_2 \leq l^*$, $u = u_1 > u_3 \geq u^*$, *and* $l_j > l_2$ *or* $u_j < u_3$ *for all* $j \in \{2, \ldots, n\}$ *and LMU-structure* (7) *that is nondecomposable and maximal is facet inducing for* P_S. □

An inequality with LMU-structure (7) is determined by three jobs and six time periods l, l_2, l^*, u^*, u_3 and u. Like in Case (1a), we have derived the exact conditions on the nine defining parameters to ensure that it is nondecomposable and maximal. Again, we refer to Van den Akker [1994] for a complete description. The number of facet inducing inequalities for P_{S^*} with structure (7) is bounded by $n^3T^4p_{\max}^2$.

Remark. It may seem more natural to define Case (1a) as $l = l_1 < l_2 < l^*$ and $u = u_1 > u_2 > u^*$, and Case (1b) as $l = l_1 < l_2 \leq l^*$ and $u = u_1 > u_3 \geq u^*$. Since under this definition Property 9 does not hold, we prefer the given one.

4.3 Case (2)

This case differs from the other ones, as the conditions on l_j and u_j and Properties 4 and 5 do not completely determine the sets L and U. It turns out to be beneficial to introduce two parameters $l' = \min\{l_j \mid j \in \{3, \ldots, n\}\}$ and $u' = \max\{u_j \mid j \in \{3, \ldots, n\}\}$, which slightly differ from l^* and u^* defined in Property 5, because it is possible that $l_2 > l'$ or $u_1 < u'$. This leads to the following property, which is similar to Property 5.

Property 10 *Let $x(V^1) + 2x(V^2) \leq 2$ be a valid, nondecomposable, and maximal inequality with $l = l_1$ and $u = u_2$.*
(a) *For all $j \in \{3, \ldots, n\}$ with $L_j \neq \emptyset$, we have $l_j = l'$ and for all $j \in \{3, \ldots, n\}$ with $L_j = \emptyset$, we have $l' - p_j \geq l$, i.e., $L_j = [l' - p_j, l]$ for all $j \in \{3, \ldots, n\}$.*
(b) *For all $j \in \{3, \ldots, n\}$ with $U_j \neq \emptyset$, we have $u_j = u'$ and for all $j \in \{3, \ldots, n\}$ with $U_j = \emptyset$, we have $u' \leq u - p_j$, i.e., $U_j = [u - p_j, u']$ for all $j \in \{3, \ldots, n\}$.* $\square$

We next investigate the structure of the set M.

Property 11 *If $x(V^1) + 2x(V^2) \leq 2$ is a valid, nondecomposable, and maximal inequality with $l = l_1$ and $u = u_2$, then*

$$\begin{aligned}
&M_1 = [u' - p_1, l'] \cap [\min\{l_2, l'\}, u - p_1] \cap [l - p_1, u_1], \\
&M_2 = [u' - p_2, l'] \cap [l, \max\{u_1, u'\} - p_2] \cap [l_2 - p_2, u], \\
&M_j = \emptyset \qquad\qquad j \in \{3, \ldots, n\}
\end{aligned}$$

Proof. Like in Case (1b), the proof of this property is analogous to that of Property 7. $\square$

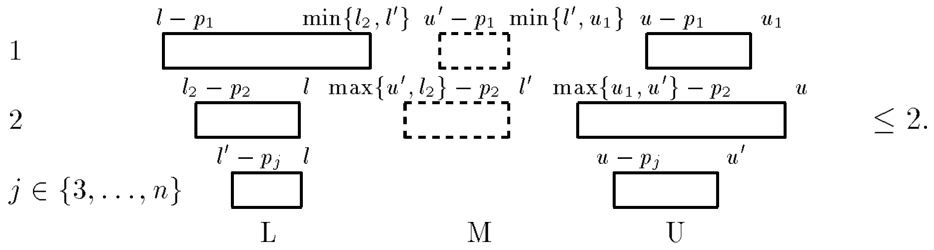
Lemma 3 and Properties 4, 10, and 11 completely determine the LMU-structure of a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1$ and $u = u_2$. We can further show that $[l' - p_2, l] \subseteq L_2$ and $[u - p_1, u'] \subseteq U_1$. We combine all these results to obtain the following theorem. Just like in the previous two cases, we have reformulated the intervals M_j to emphasize their inherent structure.

Theorem 10 *A facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1$ and $u = u_2$ has the following LMU-structure:*

$$\begin{array}{ll}
L_1 = [l - p_1, \min\{l_2, l'\}], & M_1 = [u' - p_1, \min\{l', u_1\}] \setminus (L_1 \cup U_1), \\
L_2 = [l_2 - p_2, l], & M_2 = [\max\{u', l_2\} - p_2, l'] \setminus (L_2 \cup U_2), \\
L_j = [l' - p_j, l], & M_j = \emptyset, \\
 & \\
U_1 = [u - p_1, u_1], & \\
U_2 = [\max\{u_1, u'\} - p_2, u], & \\
U_j = [u - p_j, u'] & \quad (j \in \{3, \ldots, n\}),
\end{array} \tag{8}$$

where $[l' - p_2, l] \subseteq L_2$ and $[u - p_1, u'] \subseteq U_1$. $\square$

Hence, a facet inducing inequality $x(V^1) + 2x(V^2) \leq 2$ for P_{S^*} with $l = l_1$ and $u = u_2$ can be represented by the following diagram:



The following theorem shows that the given necessary conditions are also sufficient.

Theorem 11 *A valid inequality $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1$ and $u = u_2$ and LMU-structure* (8) *that is nondecomposable and maximal is facet inducing for P_{S^*}.*

Proof. The proof proceeds along the same lines as that of Theorem 5. □

In the same way as in Case (1a), we can extend the proof of the above theorem to show that if the horizon T is large enough, then the given sufficient conditions are also sufficient for the original polytope.

Theorem 12 *If $T \geq \sum_{j=1}^{n} p_j + 5p_{\max}$ then, a valid inequality $x(V^1) + 2x(V^2) \leq 2$ with $l = l_1$ and $u = u_2$ and LMU-structure* (8) *that is nondecomposable and maximal is facet inducing for P_S.* □

An inequality with structure (8) is determined by two jobs and six time periods. Like in Case (1a), we have derived exact conditions on the parameters that ensure that the inequality is nondecomposable and maximal. The number of facet inducing inequalities with structure (8) is bounded by $2n^2T^4p_{\max}^2$.

5 Separation

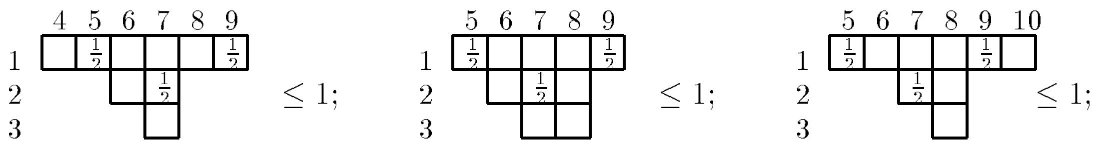
Since we want to use the classes of facet-inducing inequalities derived above in a branch-and-cut algorithm, we must solve the separation problem associated with each class, i.e., we must be able to identify an inequality in the class that cuts of the current fractional LP solution or to prove that such an inequality does not exist.

Clearly, a branch-and-cut algorithm for a single-machine scheduling problem optimizes some objective function over the convex hull of complete schedules, i.e., P_S. However, since we characterized facet inducing inequalities for P_{S^*}, i.e., the convex hull of partial schedules, our separation algorithms will identify violated inequalities for the latter polytope. Fortunately, facet inducing inequalities for P_{S^*} always define valid inequalities for P_S and, as we have shown, in many cases define facet inducing inequalities for P_S.

Our separation algorithms are based on clever enumeration. We analyze the characteristics of violated facet-inducing inequalities and use these characteristics to enumerate only a small fraction of all facet inducing inequalities while guaranteeing that a violated facet inducing inequality will be found if one exists.

We illustrate the underlying idea for the class of facet-inducing inequalities with right-hand side 1. Recall that each facet inducing inequality $x(V) \leq 1$ is completely determined by a job k, which without loss of generality is called job 1, and values l and u. Let $\tilde{x}$ be the current LP solution and let F be a subset of variables with $\tilde{x}_{jt} > 0$ for all $(j,t) \in F$ and $\tilde{x}(F) > 1$. Our separation algorithm restricts the search for a violated inequality to the subset of facet inducing inequalities covering F for which $u-l$ is minimal. A facet inducing inequality $x(V) <= 1$ covering F is minimal with respect to $u-l$ when there does not exist a facet inducing inequality $x(V') \leq 1$ with $F \subseteq V'$ and $u'-l' < u-l$, i.e., $u-l$ cannot be decreased without removing nonzero variables from the inequality. We will show that a facet inducing inequality $x(V) \leq 1$ covering F with minimal $u-l$ value has $\tilde{x}_{1,l-p_1+1} > 0$ and $\tilde{x}_{1u} > 0$. We refer to this condition as the *positive subset condition*. As a consequence of the positive subset condition, all potential violated minimal facet inducing inequalities $x(V) \leq 1$ can be enumerated in time polynomial in the number of fractional variables in the current LP solution, whereas the total number of facet inducing inequalities with right-hand side 1 is polynomial in the planning horizon T.

Example 3 Consider a three-job problem with $p_1 = 4$, $p_2 = 4$, and $p_3 = 3$. The LP solution $x_{15} = x_{19} = x_{27} = x_{2,11} = \frac{1}{2}$, $x_{31} = 1$ violates the three inequalities with structure (5) given by the diagrams below



Our separation algorithm will only examine the facet inducing inequality corresponding to the middle diagram.

The development of separation algorithms for facet inducing inequalities with right-hand side 2 is also based on the identification of positive subset conditions.

In the sequel, $\tilde{x}$ denotes the current LP-solution. As we start with the LP-relaxation of the original formulation, $\tilde{x}$ satisfies (1) and (2).

5.1 A separation algorithm for facet inducing inequalities with right-hand side 1

To identify violated facet inducing inequalities with right-hand side 1, we have to identify violated inequalities with structure (5).

The following lemma shows that the separation can be restricted to the identification of inequalities satisfying a positive subset condition which states that $\tilde{x}_{1,l-p_1+1} > 0$ and $\tilde{x}_{1u} > 0$. By this condition $u - l$ is minimal in the sense that it cannot be decreased without removing nonzero variables from the inequality.

Lemma 4 *If $\tilde{x}$ violates a facet inducing inequality $x(V) \leq 1$, then we may assume that $\tilde{x}_{1,l-p_1+1} > 0$ and $\tilde{x}_{1u} > 0$.*

Proof. Let $\tilde{x}$ violate a facet inducing inequality $x(V) \leq 1$. Since $\tilde{x}$ satisfies (4), we must have $l < u$. Suppose $\tilde{x}_{1,l-p_1+1} = 0$. If we increase l by 1, then the variable $x_{1,l-p_1+1}$ is removed from $x(V) \leq 1$ and the variables $x_{j,l+1}$ with $j \neq 1$ such that $u - p_j < l+1$ are added to this inequality. In this way we obtain another facet inducing inequality. Since in the original inequality $\tilde{x}_{1,l-p_1+1} = 0$, $\tilde{x}$ also violates the new inequality. We conclude that if $\tilde{x}_{1,l-p_1+1} = 0$, then we obtain another violated inequality by decreasing $u - l$. We may hence assume $\tilde{x}_{1,l-p_1+1} > 0$. In the same way we can show if $\tilde{x}_{1u} = 0$, then we obtain another violated inequality by decreasing u, i.e., by decreasing $u - l$. We may hence also assume that $\tilde{x}_{1u} > 0$. □

Since the current LP-solution $\tilde{x}$ satisfies the equations (3), a violated inequality $x(V) \leq 1$ must have $V_j \neq \emptyset$ for some $j \in \{2, \ldots, n\}$ and hence $u - \max\{p_j \mid j \in \{2, \ldots, n\}\} < l$.

A facet inducing inequality $x(V) \leq 1$ is determined by a job j and time periods l and u. Recall that the number of such inequalities is of order $nTp_{\max}$, where $p_{\max}$ denotes the maximal processing time. However, the number of inequalities satisfying the positive subset condition is bounded by the square of the number of fractional variables in the current LP-solution and hence the number of inequalities that have to be checked by the separation algorithm is bounded by this number. The resulting separation algorithm is as follows.

SepRHS1($\tilde{\mathbf{x}}$)

begin
  for all jobs $j \in \{1, \ldots, n\}$ **do**
    for all l such that $0 < \tilde{x}_{j,l-p_j+1} < 1$ **do**
      for all u such that $l < u < l + \max\{p_i \mid i \neq j\}$ and $0 < \tilde{x}_{ju} < 1$ **do**
        if $\sum_{s \in [l-p_j,u]} \tilde{x}_{js} + \sum_{i \neq j} \sum_{s \in [u-p_i,l]} \tilde{x}_{is} > 1$
          then violated inequality identified;
end.

5.2 A separation algorithm for facet inducing inequalities with right-hand side 2

Facet inducing inequalities with right-hand side 2 are inequalities with structure (6), (7), or (8). Because of the complexity of the necessary conditions for an inequality with one of these structures to be nondecomposable and maximal, and hence facet inducing, the separation algorithm is not restricted to facet inducing inequalities but considers all nondecomposable inequalities with one of these structures. As we have done in the previous subsection, we will study the characteristics of violated inequalities and use these characteristics to develop clever enumeration schemes. For reasons of brevity, we only consider facet inducing inequalities with structure (6) and omit proofs. The interested reader is referred to Van den Akker [1994] for additional information.

Lemma 5 *If $\tilde{x}$ satisfies all valid inequalities $x(W) \leq 1$ with $W \subseteq V$ and violates an inequality $x(V^1) + 2x(V^2) \leq 2$, then $\tilde{x}_{js} < 1$ for all $(j, s) \in V$.*

The following lemmas show that the separation can again be restricted to the identification of inequalities satisfying a positive subset condition. In this case, the positive subset condition implies that $u - l$, $u_2 - l_2$, and $(l^* - l_2)^+ + (u_2 - u^*)^+$ have to be minimal, where the expressions $(l^* - l_2)^+$ and $(u_2 - u^*)^+$ stem from the conditions on the parameters stating that $l_2 \leq l^*$ and $u_2 \geq u^*$.

Lemma 6 *If $\tilde{x}$ violates an inequality $x(V^1) + 2x(V^2) \leq 2$ with structure* (6), *then we may assume that $\tilde{x}_{1,l-p_1+1} > 0$ and $\tilde{x}_{1u} > 0$.*

Lemma 7 *If $\tilde{x}$ violates an inequality $x(V^1) + 2x(V^2) \leq 2$ with structure* (6), *then we may assume that $\tilde{x}_{2,l_2-p_2+1} > 0$, and $\tilde{x}_{2u_2} > 0$.*

Lemma 8 *If $\tilde{x}$ violates an inequality $x(V^1) + 2x(V^2) \leq 2$ with structure* (6), *then we may assume that*
(a) *if $l^* > l_2$, then either $\tilde{x}_{1l^*} > 0$, $M_1 \neq \emptyset$, and l^* is the maximum of M_1, or $\tilde{x}_{2l^*} > 0$, $M_2 \neq \emptyset$, and l^* is the maximum of M_2 ;*
(b) *if $u^* < u_2$, then either $\tilde{x}_{1u^*-p_1+1} > 0$, $M_1 \neq \emptyset$, and $u^* - p_1 + 1$ is the minimum of M_1, or $\tilde{x}_{2u^*-p_2+1} > 0$, $M_2 \neq \emptyset$, and $u^* - p_2 + 1$ is the minimum of M_2.*

Note that for an inequality $x(V^1) + 2x(V^2) \leq 2$ with structure (6) and with $l_2 < l^*$ we have that $M_1 \neq \emptyset$ and l^* is the maximum of M_1 if and only if $u^* - p_1 < l^* \leq u_2 - p_1$. The other conditions in Lemma 8 can be rewritten in a similar way.

Based on the previous lemmas, we can derive a separation algorithm for inequalities $x(V^1) + 2x(V^2) \leq 2$ with structure (6). As for facet inducing inequalities with right-hand side 1, the algorithm is based on enumeration of fractional variables in the current solution. The algorithm is more involved because when we enumerate over l^* and u^* we have to distinguish the cases $L_2 = \emptyset$, $U_2 = \emptyset$, and $L_2 \neq \emptyset \wedge U_2 \neq \emptyset$.

6 A branch-and-cut algorithm for $1|r_j|\sum w_j C_j$

Based on the separation algorithms discussed in the previous section, we have developed a branch-and-cut algorithm for the problem of minimizing the sum of the weighted completion times on a single machine subject to release dates, i.e., $1|r_j|\sum w_j C_j$, which is known to be strongly $\mathcal{NP}$-hard (Lenstra, Rinnooy Kan, and Brucker [1977]). Developing a branch-and-cut algorithm involves a lot of engineering, especially when dealing with large linear programs and large numbers of cuts. We elaborate on several such engineering aspects and show that handling them properly is of crucial importance to the overall performance of the algorithm.

The branch-and-cut algorithms have been implemented using MINTO, a Mixed INTeger Optimizer (Nemhauser, Savelsbergh, and Sigismondi [1994]). MINTO is a software system that solves mixed-integer linear programs by a branch-and-bound algorithm with linear relaxations. The user can enrich the basic algorithm by providing a variety of specialized application functions that can customize MINTO to achieve maximum efficiency for a problem class. Our computational experiments have been conducted with MINTO 2.0/CPLEX 3.0 and have been run on a IBM RS/6000 model 590.

For our computational experiments, we have used sets of 20 randomly generated instances with uniformly distributed parameters; the weights are in $[1, 10]$, the release dates are in $[0, \frac{1}{2}\sum_{j=1}^{n} p_j]$, and the processing times are in $[1, p_{\max}]$. We consider sets of 20-job instances with $p_{\max}$ equal to 5, 10, and 20, respectively, and sets of 30-job instances with $p_{\max}$ equal to 5 and 10, respectively. Recall that the number of constraints is $n + T$ and the number of variables is approximately nT. Since $T \geq \sum_{j=1}^{n} p_j$, the size of the linear program increases when the number of jobs increases as well as when the processing times increase. For the 30-job problems we did not consider $p_{\max} = 20$, since the memory requirements were too large.

6.1 Quality of the lower bounds

The goal of our first experiments was to evaluate the quality of the lower bounds obtained by just solving the LP-relaxation, by solving the LP-relaxation in combination with facet inducing inequalities with right-hand side 1, and by solving the LP-relaxation in combination with facet inducing inequalities with right-hand side 1 and 2. The results for one hundred instances, twenty in each of the sets, are summarized in Table 1. Let Z_{LB} denote a lower bound on the optimal value Z_{IP} of the integer program. The gap

G_{LB} corresponding to this lower bound is defined by

$$G_{LB} = \frac{Z_{IP} - Z_{LB}}{Z_{IP}} \times 100\%.$$

Note that this gap is expressed as a percentage. In Table 1, we report for each set of twenty instances corresponding to the same combination $(n, p_{\max})$ the following numbers:

- G_{LP}^{av} and $G_{LP}^{\max}$: the average gap after solving the LP-relaxation and the maximum of these gaps;
- G_1^{av} and $G_1^{\max}$: the average gap after the addition of cuts with right-hand side 1 and the maximum of these gaps;
- G_2^{av} and $G_2^{\max}$: the average gap after the addition of cuts with right-hand side 1 and 2 and the maximum of these gaps.

Table 1. Quality of the bounds.

	LP		1		2	
$(n, p_{\max})$	G_{LP}^{av}	$G_{LP}^{\max}$	G_1^{av}	$G_1^{\max}$	G_2^{av}	$G_2^{\max}$
(20, 5)	0.379	1.346	0.157	1.228	0.058	0.572
(20,10)	0.64	1.959	0.233	1.337	0.054	0.407
(20,20)	0.507	1.657	0.126	0.966	0.047	0.385
(30, 5)	0.390	1.309	0.179	0.664	0.121	0.599
(30,10)	0.478	1.099	0.121	0.934	0.096	0.592

These results show that the bounds obtained for these randomly generated instances are excellent, even the initial linear relaxation is always within two percent of the optimum, and that both classes of inequalities are effective in reducing the integrality gap. Table 1 indicates that for most of the instances the addition of cuts with right-hand side 1 closes at least half of the integrality gap and that addition of cuts with right-hand side 2 reduces this gap even further.

The results in Table 1 do not reflect the fact that many instances were solved to optimality just by adding cuts. Table 2 provides statistics on the frequency with which optimal solutions were found. More precisely, we report:

- n_{LP}: the number of instances for which the optimal solution of the LP-relaxation was integral;
- n_1: the total number of instances that were solved to optimality after the addition of cuts with right-hand side 1;

- n_2: the total number of instances that were solved to optimality after the addition of cuts with right-hand side 1 and 2.

Table 2. Number of instances that were solved to optimality.

$(n, p_{\max})$	n_{LP}	n_1	n_2
(20, 5)	5	12	18
(20,10)	0	6	16
(20,20)	4	13	17
(30, 5)	5	6	8
(30,10)	0	5	9

From Table 2 we conclude that the addition of cuts with right-hand side 2 significantly increases the number of instances that are solved without branching.

6.2 Branching strategies

When the addition of cuts fails to solve the problem, we resort to branch-and-bound. In this section, we discuss three branching strategies and we evaluate their performance.

In the first branching strategy, we branch on the fractional variable x_{jt} closest to 0.5 (variable dichotomy). We set $x_{jt} = 1$ on one branch, i.e., we force job j to start in time period t, and $x_{jt} = 0$ on the other branch, i.e., we prevent job j from being started in time period t. In case of ties, we select the variable with the smallest t.

In the second branching strategy, we branch on the assignment constraint $\sum_{1 \leq t \leq T-p_j+1} x_{jt} = 1$ for the job j that covers the largest time interval, i.e., the job j for which the difference between the first and last period with positive x_{jt} is maximal (GUB dichotomy). We set $\sum_{1 \leq t \leq \lfloor t^* \rfloor} x_{jt} = 1$ on one branch, i.e, we force job j to start not later than $\lfloor t^* \rfloor$, and $\sum_{\lfloor t^* \rfloor < t \leq T-p_j+1} x_{jt} = 1$ on the other branch, i.e., we force job j to start not before $\lfloor t^* \rfloor + 1$, where we choose t^* to be equal to $\sum_{1 \leq t \leq T-p_j+1} (t-1)x_{jt}$, i.e., the mean start time suggested by the current LP solution. The second branching scheme has the advantage that it divides the search space more evenly, which is a desirable characteristic of a branching strategy.

Computational experiments have revealed that these two branching strategies work best with best-bound search of the tree.

In the third branching strategy (positional branching), we exploit the structure of feasible schedules and fix jobs at certain positions in the schedule. At level d in the branch-and-bound tree the jobs in positions $1, \ldots, d-1$ have already been fixed and some job j is fixed at position d. Fixing a job j in position d is accomplished by fixing its start time at the maximum of its release date and the completion time of the $(d-1)$th job. Note that this can be done because the objective function is nondecreasing in the completion times of the jobs. As a dominance rule, we do not allow a job to be fixed in position d if its release date is so large that it is possible to complete some other job that has not yet been fixed before this release date. The subproblems at level d are defined by fixing at position d the jobs that have not been fixed yet at an earlier position. The order in which these subproblems are selected is determined on the basis of the mean start times suggested by the current LP solution, i.e., the jobs are put in nondecreasing order of $\sum_{1 \leq t \leq T-p_j+1}(t-1)x_{jt}$. This strategy works best in combination with depth-first search of the tree.

In Tables 3a and 3b, we compare the performance of the different branching strategies for the two sets of 30-job instances with $p_{\max} = 5$ and $p_{\max} = 10$. Since the majority of the 20-job instances were solved to optimality in the root node, we do not report results for these instances. In the experiments we used all cuts, i.e., cuts with right-hand side 1 as well as cuts with right-hand side 2. In the first three rows of the tables, we report on the number of nodes in the branch-and-bound tree: the average number (n^{av}), the maximum number ($n^{\max}$), and the standard deviation (σ_n). In the last three rows of the table, we report on the computation time (in seconds). Several observations

Table 3a. Performance of the different branching strategies for $n = 30$ and $p_{\max} = 5$.

(30,5)	positional branching	GUB dichotomy	variable dichotomy
n^{av}	52.30	6.60	489.90
$n^{\max}$	255.	29.	7545.
σ_n	66.02	7.63	1641.01
t^{av}	7.98	6.08	213.12
$t^{\max}$	20.31	23.99	3307.94
σ_t	5.50	4.55	716.77

can be made based on these results. First, the branching strategy based on variable dichotomy is clearly inferior to the other two. Second, GUB branching requires fewer nodes than positional branching. However, evaluating fewer nodes does not translate into faster solution times. There are two factors that, in our opinion, contribute to this phenomenon. Positional branching fixes many more variables, which reduces the size of the linear programs that have to be solved. In addition, in a depth-first search strategy consecutive linear programs differ only slightly. Consequently, the basis of last solved linear program is a good starting basis for the current linear program. In a best-bound search strategy consecutive linear programs are likely to differ considerably. Consequently, the basis of the last solved linear program does not provide a good starting

Table 3b. Performance of the different branching strategies for $n = 30$ and $p_{\max} = 10$.

(30,10)	positional branching	GUB dichotomy	variable dichotomy
n^{av}	26.57	19.35	169.05
$n^{\max}$	286.	247.	2661.
σ_n	56.62	52.86	578.68
t^{av}	23.27	53.15	477.38
$t^{\max}$	384.63	691.51	7269.59
σ_t	60.10	146.96	1585.95

basis for the current linear program. Furthermore, since many cuts are generated during the solution process, the basis associated with the last linear program solved in the parent node does not provide a good starting basis either. A final observation is that there is a high variation in complexity among the instances. With GUB branching all but one instance are solved in fewer than 20 nodes and less than 60 seconds; the one difficult instance took a little less than 250 nodes and 700 seconds. To verify whether this is typical behavior, we generated 20 additional 30-job instances with $p_{\max} = 10$ and tested GUB branching and positional branching on the extended set of 40 instances. The results for this extended set of instances can be found in Table 4 and show a similar pattern.

Table 4. Performance of the different branching strategies for $n = 30$ and $p_{\max} = 10$.

(30,10)	positional branching	GUB dichotomy
n^{av}	133.38	29.05
$n^{\max}$	2108.	573.
σ_n	370.06	95.92
t^{av}	83.49	59.98
$t^{\max}$	638.79	691.51
σ_t	158.91	142.37

An advantage of the branching strategy based on GUB dichotomy is that it can be applied for all objective functions $\sum_{j=1}^{n}\sum_{t=1}^{T-p_j+1} c_{jt}x_{jt}$, whereas the positional branching strategy is based on the assumption that it is most favorable to start a job as early as possible, i.e., it can only be applied if the objective function is nondecreasing in the completion times of the jobs.

On the other hand, with the positional branching strategy the total number of nodes in the branch-and-bound tree only depends on the number of jobs, whereas for the branching strategy based on GUB dichotomy the number of nodes depends on the number of jobs as well as on the planning horizon, i.e., on the size of the processing times. This suggests that positional branching may perform better for instances with large processing times.

6.3 Cut generation schemes

In this subsection, we study the influence of different cut generation schemes on the performance of the branch-and-cut algorithm. Cut generation schemes try to find the proper balance between the expected increase in performance due to stronger bounds that result from the addition of cuts and the expected decrease in performance due to the effort required to identify violated cuts and to solve larger and more difficult linear programs. Cut generation schemes specify, among other things, when we try to identify violated inequalities, which of the identified violated inequalities are added, and when inactive inequalities are deleted.

The experiments of the previous section showed that 30-job instances with $p_{\max} = 5$ are relatively easy, in the sense that their solution requires very few nodes, and that a large sample of 30-job instances with $p_{\max} = 10$ is necessary to be able to draw reliable conclusions. Therefore, the remaining experiments have been conducted on the extended set of 40 randomly generated 30-job instances with $p_{\max} = 10$.

We have investigated various possible cut generation schemes that specify choices related to which classes of cuts to use and when to use them.

R12T12: At all nodes, add cuts with right-hand side 1 and 2.

R12T1: At the root node, add cuts with right-hand side 1 and 2; in all other nodes, add cuts with right-hand side 1.

R12: At the root node, add cuts with right-hand side 1 and 2; in all other nodes, do not add cuts.

R1T1: At all nodes, add cuts with right-hand side 1.

The performance of these variants is shown in Table 5a and 5b. We report the performance of these variants with positional branching as well as with GUB branching. Again, n^{av} and $n^{\max}$ denote the average and maximum number of nodes, and t^{av} and $t^{\max}$ denote the average and maximum computation time (in seconds).

Several observations can be made based on these results. First, it is advantageous to generate cuts throughout the search tree. Second, the cut generation scheme *R12T12*,

Table 5a. Cut generation schemes with positional branching.

	R12T12	*R12T1*	*R12*	*R1T1*
n^{av}	133.38	220.90	377.07	422.07
$n^{\max}$	2108	3677	7504	4764
t^{av}	83.49	71.25	75.62	64.01
$t^{\max}$	638.79	528.46	816.73	595.59

Table 5b. Cut generation schemes with GUB branching.

	R12T12	*R12T1*	*R12*	*R1T1*
n^{av}	29.05	67.42	107.70	89.97
$n^{\max}$	573	1981	3443	1595
t^{av}	59.98	69.54	66.09	59.81
$t^{\max}$	691.51	889.87	991.02	846.20

i.e., generating cuts with right-hand side 1 and 2, clearly results in the fewest number of evaluated nodes. However, evaluating fewer nodes does not translate into faster solution times. For positional branching, cut generation scheme *R1T1*, i.e., generating only cuts with right-hand side 1, is much faster than *R12T12* even though it generates considerably more nodes, and for GUB branching, cut generation scheme *R1T1* is about as fast as *R12T12* although it generates more nodes. This is probably due to the fact that the linear programs that result if cuts with right-hand side 2 are added are more difficult because they are denser than the ones resulting from the addition of cuts with right-hand side 1. So far the two best variants of the algorithm are positional branching with cut generation scheme *R1T1* and GUB branching with cut generation scheme *R12T12*. We prefer cut generation scheme *R12T12* over *R1T1* for GUB branching because it seems to be more robust in the sense that the maximum number of evaluated nodes and the maximum computation time over all instances are the smallest. For the remainder, we will restrict our computational experiments to these two variants.

The cut generation schemes discussed above specify choices related to which classes of cuts to use and when to use them. We have also considered cut generation schemes that try to improve the performance by limiting the number of violated inequalities that will be added to the active linear program. In fact, such a cut generation scheme has been active during all previous experiments. When MINTO processes a node, it monitors the changes in the value of the LP solutions from iteration to iteration. If it detects that the total change in the value of the LP solution in the last three iterations is less than 0.5 percent, i.e., 0.005 times the value of the current LP solution, it forces MINTO to branch. This feature is incorporated in MINTO to handle the 'tailing-off' effect exhibited by many cutting plane algorithms.

The experiments carried out to evaluate the quality of the bounds, discussed in Section 6.1, revealed that it is impossible to predict the change in objective function value after the addition of violated inequalities. It frequently happened that the objective function hardly changed for several iterations before improving significantly. Consequently, it is very likely that MINTO, with default settings, would sometimes force branching too soon. To ensure the best possible bound at the root node, we have chosen to deactivate forced branching in the root node.

To evaluate the effect of different forcing strategies on the performance of the algorithms, we have investigated the following three strategies: no forced branching, no forced branching at the root node but forced branching at all other nodes, and forced branching throughout the tree. The results are shown in Tables 6a and 6b. We conclude

Table 6a. Forcing strategies with GUB branching.

	no forcing	no root forcing	forcing
n^{av}	27.92	29.05	56.12
$n^{\max}$	419	573	1419
t^{av}	130.33	59.98	56.45
$t^{\max}$	2396.48	691.51	804.85

Table 6b. Forcing strategies with positional branching.

	no forcing	no root forcing	forcing
n^{av}	432.70	422.07	409.80
$n^{\max}$	5036	4746	4746
t^{av}	65.55	64.01	62.32
$t^{\max}$	602.62	595.59	608.17

the following from these results. First, the tailing-off effect is much stronger when cuts with right-hand side 2 are used. Second, the strategy that we adopted, i.e., no forcing at the root node, works well.

There are various other ways to limit the number of violated inequalities that will be added to the active linear program: limit the number of cuts that are added in a single round of cut generation, limit the number of rounds of cut generation per node evaluation, and limit the number of nodes at which cut generation takes place (this is sometimes referred to as the cut frequency). All these did not seem to have a significant positive effect on the performance of the basic algorithm. In most cases, the performance of these variants was actually worse.

Finally, we have experimented with cut generation schemes in which inequalities are deleted when they have been inactive for a number of consecutive iterations, i.e., the dual variable associated with the inequality has been 0 for a number of consecutive iterations. Table 7 shows the effect of different thresholds for deletion (10, 50, 1000) for GUB branching. Note that setting the threshold to 1000 for this application is equivalent to no cut deletion. We see that cut deletion does influence the computation time and

Table 7. Effect of different thresholds for cut deletion.

	10	50	1000
n^{av}	28.82	29.05	30.42
$n^{\max}$	573	573	595
t^{av}	75.98	59.98	80.06
$t^{\max}$	901.98	691.51	1320.93

that a threshold of 50 seems appropriate for our application. We have not performed the same experiment for positional branching, but it is very likely that our conclusions are also applicable to positional branching.

6.4 Primal heuristics

In this subsection, we describe the primal heuristics that have been incorporated in the branch-and-cut algorithm. The availability of good feasible solutions is important for various reasons. In case of depth-first search (which we do in case of positional branching) it may significantly reduce the number of nodes that have to be evaluated, since any node with a lower bound greater than or equal to the value of the best known solution can be skipped from further consideration. In case of best-bound search (which we do in case of GUB branching) it will not reduce the number of evaluated nodes by much, but it will reduce the set of unevaluated nodes that has to be kept, which is important for large integer programs because it reduces the chance of running out of memory. Furthermore, good feasible solutions are essential for effective reduced cost fixing.

We have implemented four primal heuristics. The first heuristic is derived from Smith's rule (Smith [1956]). Smith's rule solves $1||\sum w_jC_j$, i.e., the case without release dates. Smith's rule states that $1||\sum w_jC_j$ is solved by scheduling the jobs in order of nondecreasing p_j/w_j ratio. Our first heuristic schedules the jobs according to the following rule: at each decision point schedule the available job with the smallest p_j/w_j

ratio, where the first decision point is the smallest release date, and the kth decision point is either the completion time of the job scheduled in the $(k-1)$th position or, in case there are no jobs available at that time, the smallest release date among the unscheduled jobs.

The other three heuristics schedule the jobs according to some ordering based on the values of the current linear programming solution. We have used the following three orderings:

- schedule jobs in order of nondecreasing mean start time $\sum_{t=1}^{T-p_j+1}(t-1)x_{jt}$;
- schedule jobs in order of nondecreasing maximum start time $argmax_t\{x_{jt}\}$;
- schedule jobs in order of nondecreasing first start time $argmin_t\{x_{jt} > 0\}$.

In most situations, the ordering based on the mean start time provides the best feasible solution. However, since these heuristics take very little time we always apply all of them. Furthermore, note that these heuristics are applied every time that a linear program has been solved, whereas the first heuristic is applied only once.

Let z_{UB} denote an upper bound on the optimal value z_{IP} of the integer program. The gap G_{UB} corresponding to this upper bound is defined by

$$G_{UB} = \frac{z_{UB} - z_{IP}}{z_{IP}} \times 100\%.$$

In Table 8, we report for those 30-job instances that were not solved to optimality by the initial LP-relaxation the following numbers:

- $G_{\text{ratio}}^{\text{av}}$ and $G_{\text{ratio}}^{\text{max}}$: the average gap for the first heuristic and the maximum of these gaps;
- $G_{\text{init}}^{\text{av}}$ and $G_{\text{init}}^{\text{max}}$: the average gap for the best of the other three heuristics when applied to the solution of the initial LP relaxation and the maximum of these gaps;
- $G_{\text{root}}^{\text{av}}$ and $G_{\text{root}}^{\text{max}}$: the average gap after the root node has been evaluated and the maximum of these gaps.

Observe that the gap after the root node has been evaluated may differ for the two variants we consider, since we do not generate cuts with right-hand side 2 with the positional branching scheme.

The computational results show that the solutions to the LP-relaxations encountered during the solution process provide good starting-points for obtaining primal solutions; the heuristics based on these fractional solutions provide much better primal solutions than the first heuristic. Recent results on approximation algorithms for machine scheduling problems [Goemans 1997, Hall, Schulz, Shmoys, and Wein 1997] provide theoretical evidence of the strength of LP-based heuristics for single machine scheduling problems.

Table 8. Performance of the primal heuristics.

				$RHS1$		$RHS12$	
$G^{\text{av}}_{\text{ratio}}$	$G^{\text{max}}_{\text{ratio}}$	$G^{\text{av}}_{\text{init}}$	$G^{\text{max}}_{\text{init}}$	$G^{\text{av}}_{\text{root}}$	$G^{\text{max}}_{\text{root}}$	$G^{\text{av}}_{\text{root}}$	$G^{\text{max}}_{\text{root}}$
9.03	17.52	1.47	6.94	0.44	2.22	0.19	1.23

7 Related research and conclusions

As mentioned in the introduction, Sousa and Wolsey [1992] and Crama and Spieksma [1993] have also studied the time-indexed formulation of single machine scheduling problems. In this section, we briefly indicate the relation between their research and our research.

Sousa and Wolsey present three classes of valid inequalities. The first class consists of inequalities with right-hand side 1, and the second and third class consist of inequalities with right-hand side $k \in \{2, \ldots, n-1\}$. Each class of inequalities is derived by considering a set of jobs and a certain time period. The right-hand side of the resulting inequality is equal to the cardinality of the considered set of jobs.

Sousa and Wolsey show that the inequalities in the first class, which is exactly the class of inequalities with structure (5), are all facet inducing, if $T \geq \sum_{j+1}^{n} p_j + 3p_{\max}$. In Section 3, we have complemented this result by showing that *all* facet inducing inequalities with right-hand side 1 for the extended polytope P_{S^*} are in this class, and hence all facet inducing inequalities with right-hand side 1 for the original polytope P_S have a representation in this class.

With respect to the other two classes of valid inequalities studied by Sousa and Wolsey we make the following observations. Any inequality in the second class that has right-hand side 2 can be lifted to an inequality with LMU-structure (6) if $p_{k_1} \neq p_{k_2}$, and to an inequality with LMU-structure (8) if $p_{k_1} = p_{k_2}$, where $\{k_1, k_2\}$ is the set of jobs considered. Any inequality in the third class that has right-hand side 2 can be written as the sum of two valid inequalities with right-hand side 1.

Sousa and Wolsey also developed a cutting plane algorithm based on the three classes of inequalities they derived. We have only been able to compare our algorithm with their algorithm on a set of 4 instances. Each one is solved at the root node by both algorithms. Therefore, we cannot make any meaningful comparative statements.

Crama and Spieksma investigate the special case of equal processing times. They completely characterize all facet inducing inequalities with right-hand side 1 and present two other classes of facet inducing inequalities with right-hand side $k \in \{2, \ldots, n-1\}$.

Our characterization of all facet inducing inequalities with right-hand side 1 was found independently and generalizes their result. The inequalities in their second class that have right-hand side 2 are special cases of the inequalities with LMU-structure (8), and the inequalities in their third class that have right-hand side 2 are special cases of the inequalities with LMU-structure (6). In addition to the facet inducing inequalities reported in their paper, they have identified other classes of facet inducing inequalities with right-hand side 2 [Spieksma 1991].

Crama and Spieksma also developed a branch-and-cut algorithm based on the classes of facet-inducing inequalities they derived. They tested their algorithm on two classes of problems. The first one has randomly generated objective coefficients c_{jt}. The second one has objective coefficients $c_{jt} = w_j(t - r_j)$ if $r_j \leq t \leq d_j$ and $c_{jt} = M$ otherwise, where M is some large integer; these instances model minimization of the weighted sum of the completion times subject to release dates and deadlines, where release dates and deadlines may be violated at large cost. For both problem classes the performance of our algorithm and their algorithm is comparable. However, their branch-and-cut algorithm incorporates classes of cuts that have been derived specifically for problems with equal processing times, whereas our algorithm does not.

For the problem $1|r_j|\sum w_j C_j$, several combinatorial branch-and-bound algorithms have been developed, i.e., branch-and-bound algorithms that are not based on linear programming relaxations. An example is the algorithm of Belouadah, Posner, and Potts [1992]. The lower bounds in their algorithm are based on job-splitting. The number of nodes that have to be evaluated by their algorithm is larger than the number of nodes that have to be evaluated by our algorithm, but their algorithm requires less computation time. This indicates that our lower bounds are better, but that we need more time to compute them. This is due to the fact that we have to solve large linear programs. However, our branch-and-cut algorithm can easily be applied to many types of scheduling problems with various objective functions, whereas these combinatorial branch-and-bound algorithms are typically designed for one specific problem type.

We conclude that the strength of the presented branch-and-cut algorithm is that it can be applied successfully to a wide range of single-machine scheduling problems, but that its weakness is the fact that in its current form it is limited to instances with a relatively small number of jobs and relatively small processing times, because otherwise the time to solve the linear programs becomes prohibitive. In a sequel paper (Van den Akker, Hurkens, and Savelsbergh [1995]), we will investigate column generation as a way of handling this weakness.

The new classes of facet inducing inequalities that we have derived and subsequently incorporated in a branch-and-cut algorithm are valuable, since they reduce the integrality gap and have allowed us to solve larger instances, in terms of processing times, than have been solved with other branch-and-cut codes.

Another important strength of the proposed approach is the quality of the feasible solutions obtained at the root node. The embedded LP-based heuristics produce high quality feasible solutions.

Acknowledgment

The authors wish to thank Cor Hurkens for his useful remarks and suggestions and for his help with the computational experiments and an anonymous referee for his comments on an earlier draft of the paper.

References

J.M. van den Akker (1994). *LP-based solution methods for single-machine scheduling problems.* PhD Thesis, Eindhoven University of Technology.

J.M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh (1995). *Column generation for single-machine scheduling problems.* In preparation.

E. Balas (1985). On the facial structure of scheduling polyhedra. *Mathematical Programming Study 24*, 179-218.

H. Belouadah, M.E. Posner, and C.N. Potts (1992). Scheduling with release dates on a single machine to minimize total weighted completion time. *Discrete Applied Mathematics 36*, 213-231.

Y. Crama and F.C.R. Spieksma (1993). *Scheduling jobs of equal length: complexity and facets.* Report M 93-09, Faculty of Economics, University of Limburg, Maastricht.

CPLEX Optimization, Inc. (1994). *Using the CPLEX Callable Library, Version 3.0.*

H. Crowder, E.L. Johnson, M.W. Padberg (1983). Solving large-scale zero-one linear programming problems. *Operations Research 31*, 803-834.

M.E. Dyer, L.A. Wolsey (1990). Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics 26*, 255-270.

M.X. Goemans (1997). Improved approximation algorithms for scheduling with release dates. To appear in *Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms.*

L.A. Hall, A.S. Schulz, D.B. Shmoys, J. Wein (1997). Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. To appear in *Mathematics of Operations Research.*

P.L. Hammer, E.L. Johnson, and U.N. Peled (1975) Facets of regular 0-1 polytopes. *Mathematical programming 8*, 179-206.

K.L. Hoffman, M.W. Padberg (1985). LP-based combinatorial problem solving. *Annals of Operations Research 4*, 145-194.

J.B. Lasserre, M. Queyranne (1992). Generic scheduling polyhedra and a new mixed-integer formulation for single-machine scheduling. E. Balas, G. Cornuéjols, R. Kannan) (eds.). *Integer Programming and Combinatorial Optimization*, Proceedings of the IPCO-Conference held at Carnegie Mellon University, University Printing and Publications, Carnegie Mellon University, Pittsburgh.

E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys (1993). Sequencing and scheduling: Algorithms and complexity, in: *Handbooks in Operations Research and Management Science, Vol. 4,* S.C. Graves et al. (ed.), North-Holland, Amsterdam, 445-522.

J.K. Lenstra, A.H.G. Rinnooy Kan, and P. Brucker (1977). Complexity of machine scheduling problems. *Annals of Discrete Mathematics 1*, 343-362.

G.L. Nemhauser, M.W.P. Savelsbergh (1992). A cutting plane algorithm for the single machine scheduling problem with release times. M. Akgül, H. Hamacher, S. Tufecki (eds.). *Combinatorial Optimization: New Frontiers in the Theory and Practice*, NATO ASI Series F: Computer and Systems Sciences 82, Springer, Berlin, 63-84.

G.L. Nemhauser, M.W.P Savelsbergh, and G.C. Sigismondi (1994). MINTO, a Mixed INTeger Optimizer. *Operations Research Letters 15*, 47-58.

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

M.W. Padberg, G. Rinaldi (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review 33*, 60-100.

M. Queyranne (1993). Structure of a simple scheduling polyhedron. *Mathematical Programming 58*, 89-110.

M. Queyranne and A.S. Schulz (1997). Polyhedral Approaches to Machine Scheduling. To appear in *Mathematical Programming B.*

M. Queyranne, Y. Wang (1991). Single machine scheduling polyhedra with precedence constraints. *Mathematics of Operations Research 16*, 1-20.

T.J. van Roy and L.A. Wolsey (1987). Solving Mixed 0-1 Programs by Automatic Reformulation, *Operations Research 35*, 45-57.

W.E. Smith (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly 3*, 59-66.

J.P. de Sousa (1989). *Time-indexed formulations of non-preemptive single-machine scheduling problems.* Ph.D. Thesis, Catholic University of Louvain, Louvain-la-Neuve.

J.P. de Sousa and L.A. Wolsey (1992). A time-indexed formulation of non-preemptive single-machine scheduling problems. *Mathematical Programming 54*, 353-367.

F.C.R. Spieksma (1991). Personal communication.

L.A. Wolsey (1989). *Formulating single machine scheduling problems with precedence constraints.* CORE discussion paper 8924, Catholic University of Louvain, Louvain-la-Neuve.