Polyhedral Approaches to Machine Scheduling *

Maurice Queyranne
University of British Columbia
Faculty of Commerce
2053 Main Mall
Vancouver, B. C.
Canada V6T 1Z2
Maurice.Queyranne@ubc.ca

Andreas S. Schulz
Technische Universität Berlin
Fachbereich Mathematik (MA 6–1)
Straße des 17. Juni 136
D–10623 Berlin
Germany
schulz@math.tu–berlin.de

November 1994
Revised, October 1996

Abstract

We provide a review and synthesis of polyhedral approaches to machine scheduling problems. The choice of decision variables is the prime determinant of various formulations for such problems. Constraints, such as facet inducing inequalities for corresponding polyhedra, are often needed, in addition to those just required for the validity of the initial formulation, in order to obtain useful lower bounds and structural insights. We review formulations based on time–indexed variables; on linear ordering, start time and completion time variables; on assignment and positional date variables; and on traveling salesman variables. We point out relationship between various models, and provide a number of new results, as well as simplified new proofs of known results. In particular, we emphasize the important role that supermodular polyhedra and greedy algorithms play in many formulations and we analyze the strength of the lower and upper bounds obtained from different formulations and relaxations. We discuss separation algorithms for several classes of inequalities, and their potential applicability in generating cutting planes for the practical solution of such scheduling problems. We also review some recent results on approximation algorithms based on some of these formulations.

1 Introduction

Polyhedral combinatorics is a well established approach to combinatorial optimization problems (cf., e.g., Pulleyblank [Pul89]). It may often provide rich structural results and insights, and lead to new exact or approximate solution methods. By associating points (incidence vectors) in an Euclidean space with the feasible solutions to an instance of the combinatorial optimization problem and by studying the convex hull of these points – particularly in order to obtain linear inequalities describing it – we can apply polyhedral theory and linear programming techniques. The success of this approach depends highly on the choice of variables, a question typically addressed first when formulating a model. This choice

*This work was done in part during mutual visits of the authors to each other's institution. This research was supported in part by a research grant of the Natural Sciences and Engineering Research Council (NSERC) of Canada to the first author, and by the graduate school "Algorithmische Diskrete Mathematik". The latter is supported by the Deutsche Forschungsgemeinschaft, grant We 1265/2-1.

not only influences the structure of the associated polyhedron but also determines whether and how certain objective functions and (additional) restrictions can be modeled by means of linear equations and inequalities. To compare different approaches arising from different choices of variables, the quality of the bounds obtained from the corresponding LP relaxations, the number of variables needed, and the number of constraints have to be taken into account. When solving hard problems, a large number of variables can often be handled efficiently if the corresponding pricing problem is well solved, exactly or approximately. Similarly, a large number of constraints can be handled efficiently if the corresponding separation problem is well solved. In contrast, poor lower bounds often make exact solution methods impractical. In the *branch&cut* method, one starts with an LP relaxation, and then successively adds inequalities that are valid for all feasible points but not satisfied by the current optimal solution to the LP relaxation. For this, at each iteration one solves for this current LP solution the separation problem associated with the convex hull of feasible points. Unless the current solution is feasible to the original problem (in which case we are done), the separation subroutine produces one or several violated inequalities (cutting planes). The optimal values of the LP relaxations provide increasingly better lower bounds on the value of the minimal solution of the combinatorial optimization problem. Branching occurs only when no violated inequalities are found (in reasonable time) to cut off infeasible solutions. (This is possible since the separation problem is equivalent to the optimization problem with respect to polynomial time solvability, see Grötschel, Lovász and Schrijver [GLS88]. We also refer to [GLS88] for a comprehensive treatment of all theoretical aspects concerning the importance and usefulness of polyhedral combinatorics.) Hence, two important theoretical and practical issues are to determine good (partial) descriptions of the convex hull of feasible points in terms of linear equations and inequalities, and to solve the separation problem associated with the obtained classes of inequalities efficiently.

The general purpose of the present paper is to review and glue together the results that were obtained by applying the polyhedral approach to scheduling problems, to complement some of these results and provide simplified proofs of other, to present new insights, and to point out relationships between various approaches.

Most research in this vein has been done for nonpreemptive single machine scheduling. Even Balas [Bal85] who pioneered the work on scheduling polyhedra by considering problems of job shop type mainly concentrated on the single machine. Consequently, we will focus on single machine scheduling. But we also outline possible extensions to multiple machine scheduling.

The paper is divided into six sections. In this first introductory section we present the required terminology and notation, and outline some fundamental results on polyhedra and supermodular systems. Each of the remaining sections is devoted to one possible choice of variables. Since nonpreemptive schedules are completely determined by the job completion times C_j or, equivalently, the start times S_j, several authors including Balas [Bal85], Queyranne [Que93], Queyranne and Wang [QW91a, QW91b], von Arnim and Schrader [AS93], and von Arnim and Schulz [AS94] studied the convex hull of feasible completion time vectors. This work is presented in a unifying way in Section 2. Time is handled more implicitly if $0/1$ variables x_{jt} are used that are not only indexed by jobs but additionally by time periods. Section 3 investigates polytopes arising from the use of these variables, based on earlier work of Sousa and Wolsey [SW92], Crama and Spieksma [CS95], and van den Akker, van Hoesel, and Savelsbergh [AHS93]. In Section 4 we do not care about time, and just want to know for each pair of jobs j,k which precedes the other. Hence, we obtain linear ordering variables δ_{jk}. These are one if j precedes k and zero, otherwise. Linear ordering variables have been used, among others, by Peters [Pet88], Dyer and Wolsey [DW90], Wolsey [Wol90a], and Nemhauser and Savelsbergh [NS92]. Positional date variables τ_κ and γ_κ denoting the start time and completion time of the κ–th job in the schedule, respectively, have been used together with assignment variables $u_{j\kappa}$ by Lasserre and Queyranne [LQ92]. In Section 5 we summarize their results from a different perspective and present simplified proofs for most of them. The final sixth section deals with problems including sequence–dependent processing times which lead immediately to traveling salesman variables z_{jk}.

1.1 Preliminaries

In machine scheduling the resources, that is, the machines (or processors) are assumed to be unable to process more than one job (task) at a time. In addition, a job requires at most one machine at a time. If we consider multiple machine problems, m always denotes the number of machines, and i $(i = 1,\dots,m)$ is used to refer to a special machine. Throughout this paper, N will be the set of jobs to be processed, and we assume that $N = \{1,\dots,n\}$. Unless otherwise stated, each job requires uninterrupted processing, that is, preemption is not allowed. In referring to jobs we use the symbols $j,k,$ and ℓ. There may be different data associated with each job, and we can always assume, w. l. o. g., that these are integral. The processing time of job j is denoted by $p_j > 0$; w_j is its weight when dealing with objective functions including weights; r_j is the time job j becomes available for processing (and assumed to be zero unless otherwise stated); and $\bar{d}_j$ stands for the time at which job j has to be finished. To avoid confusion, we use d_j for due dates. When there are precedence constraints between the jobs we may see them as a partial order on the set of jobs. Throughout, we represent a partially ordered set (poset, for short) as a digraph $D = (N,A)$. Clearly, this digraph is acyclic and transitively closed. An arc (j,k) indicates that job k cannot be started before job j has been completed. A *schedule* is an allocation of one or more time intervals on one or more machines to each job such that no two time intervals on the same machine overlap, such that no time intervals allocated to the same job overlap, and such that specific requirements concerning the machine environment and the job characteristics, e. g., release dates, deadlines, or precedence constraints, are met. Whenever we discuss a certain scheduling model we describe it in detail but refer, for convenience, also to the standard classification scheme of scheduling problems [GLLRK79]. The reader who is more interested in the various scheduling models and algorithms should, for instance, consult the survey of Lawler, Lenstra, Rinnooy Kan, and Shmoys [LLRKS93].

1.1.1 Polyhedral Theory

For a detailed treatment of polyhedral theory we refer the reader to Schrijver [Sch86] and Nemhauser and Wolsey [NW88], for the concepts needed in polyhedral combinatorics especially to Pulleyblank [Pul89]. Here, we briefly summarize some basic concepts and results.

Throughout this paper we deal with objects in the real vector space $\mathbb{R}^E$ where the components of each vector $x = (x_e)_{e\in E}$ are indexed by the members of a finite set E, where E can be the set of jobs in a scheduling problem, the set of edges in a graph, and so on. Here and whenever convenient, we ignore the artificial distinction between vectors with components subscripted by elements of the finite set E on the one hand and mappings from E to $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, or $\mathbb{R}$ on the other hand. We let $\mathbb{N}^E$, $\mathbb{Z}^E$, $\mathbb{Q}^E$, and $\mathbb{R}^E$ denote the set of these objects, respectively, and write $c(S)$ for $\sum_{u\in S} c_u$, when $S \subseteq E$ and $c \in \mathbb{R}^E$. Given a subset $S \subseteq E$, we denote by $\chi^S \in \{0,1\}^E$ the *characteristic vector* (*function*) of S, i. e., $\chi^S_e = 1$ if $e \in S$ and $\chi^S_e = 0$, otherwise. We denote by $\mathbb{1}$ the all–one vector of suitable dimension.

A *polyhedral cone* C is a cone of the form $C = \{x \in \mathbb{R}^E : Ax \leqslant 0\}$ for some matrix A, i. e., C is the intersection of finitely many closed linear halfspaces. A *polyhedron* P in $\mathbb{R}^E$ is the intersection of a finite number of closed halfspaces, i. e., $P = \{x \in \mathbb{R}^E : Ax \leqslant b\}$ for a matrix A and a vector b. Equivalently, polyhedra can be defined as the Minkowski sum of the convex hull of finitely many points and the conical hull of finitely many directions, $P = \operatorname{conv}(V) + \operatorname{cone}(R)$. The latter description is useful for defining polyhedra associated with combinatorial optimization problems. A *polytope* is a bounded polyhedron. Thus, a polytope can also be generated as the convex hull of finitely many points. An inequality $ax \leqslant \beta$ is *valid* for the polyhedron P if P is contained in the halfspace induced by this inequality. The intersection of P with the corresponding hyperplane $\{x \in \mathbb{R}^E : ax = \beta\}$ is called a *face* of P. *Facets* are the maximal faces of a polyhedron which are distinct from the polyhedron itself. The *dimension* of a polyhedron P is the dimension of its affine hull aff(P).

Lemma 1.1. *Let $\{x \in \mathbb{R}^E : Dx = d\}$ be the affine hull of the polyhedron P. Then*

$$\dim(P) \quad = \quad |E| - \operatorname{rank}(D) \ .$$

If D is a matrix of full row rank such that aff$(P) = \{x \in \mathbb{R}^E : Dx = d\}$, we call $Dx = d$ a *minimal equation system* for P. Recall that P is q–dimensional if and only if P contains exactly $q+1$ affinely independent points.

Given a polyhedron in one of the two forms mentioned above – as the intersection of finitely many halfspaces or as the Minkowski sum of a polytope and a polyhedral cone – the question arises whether these descriptions are minimal. First, let $P = \{x \in \mathbb{R}^E : Dx = d,\ Ax \leqslant b\} \neq \emptyset$. The system $Dx = d$, $Ax \leqslant b$ is *irredundant* (or minimal) with respect to P

- if $Ax \leqslant b$ does not contain any implicit equations, i. e., there exists a point $x \in P$ such that $Ax < b$;
- if D has full row rank, i. e., $Dx = d$ is a minimal equation system; and
- if no inequality of $Ax \leqslant b$ is implied by the other constraints.

If $Dx = d$, $Ax \leqslant b$ is irredundant, then each inequality of $Ax \leqslant b$ induces a facet of P and each facet of P is induced by exactly one of these inequalities. Hence, if we want to describe a polyhedron in terms of linear inequalities, facet inducing inequalities are the best we can look for. The following theorem provides alternative characterizations of facets.

Theorem 1.2. *Let F be a face of the polyhedron $P, \emptyset \subset F \subset P$, and let the affine hull of P be the solution set of $Dx = d$. Then the following statements are equivalent:*

(a) *F is a facet of P.*

(b) $\dim(F) = \dim(P) - 1$.

(c) *If $ax \leqslant \beta$ and $cx \leqslant \delta$ are valid for P with $F = \{x \in P : ax = \beta\} = \{x \in P : cx = \delta\}$, then there exist a vector λ and a scalar $\mu > 0$ such that $a = \mu c + \lambda D$ and $\beta = \mu\delta + \lambda d$.*

From Theorem 1.2 follows in particular that the facet inducing inequalities of a full–dimensional polyhedron are unique, up to positive multiples. A valid inequality $ax \leqslant \beta$ for a polyhedron P is *dominated* by a valid inequality $cx \leqslant \delta$ if $\{x \in P : ax = \beta\} \subseteq \{x \in P : cx = \delta\}$.

Now, assume that a polyhedron P is given by its internal representation. For simplicity, we assume P to be pointed, i. e., P has at least one extreme point (vertex). Vertices can be characterized as follows.

Theorem 1.3. *Let P be a polyhedron, and $x \in P$. Then the following statements are equivalent:*

(a) *x is a vertex of P.*

(b) *The set $\{x\}$ is a face of P of dimension zero.*

(c) *For all points* $y, z \in P$ *distinct from* x *and all* $0 < \lambda < 1$ *we have* $x \neq \lambda y + (1-\lambda)z$.

(d) *There exists a vector* c *such that* x *is the unique optimal solution of the linear programming problem* $\min\{cy : y \in P\}$.

The vertices and extreme rays play the same role in the internal representation as the facets in the external one. That is, P is the Minkowski sum of the convex hull of its vertices and the conical hull of its extreme rays.

A *ray* of a cone C is the set of all nonnegative multiples of some $y \in C$, called the direction of the ray. A vector $y \in C$ is *extreme*, if for any $y_1, y_2 \in C$, $y = \frac{1}{2}(y_1 + y_2)$ implies that y_1 and y_2 belong to the ray generated by y. A ray is extreme if its direction vector is extreme.

The *recession cone* $\mathrm{rec}(P)$ of P is the set of its directions to "infinity". Thus, if $P = \mathrm{conv}(V) + \mathrm{cone}(R) = \{x : Ax \leqslant b\}$, then $\mathrm{rec}(P) = \mathrm{cone}(R) = \{y : Ay \leqslant 0\}$.

Linear programming is concerned with problems of the type

$$\begin{aligned} \text{maximize} \quad & cx \\ \text{subject to} \quad & Ax \leqslant b \\ & x \in \mathbb{R}^E_+, \end{aligned} \tag{1.1}$$

where $A \in \mathbb{R}^{M \times E}$ is a given matrix and $b \in \mathbb{R}^M$ and $c \in \mathbb{R}^E$ are given vectors of suitable dimension, respectively. *Duality* deals with pairs of linear programming problems and the relationship between their solutions. The *dual* to (1.1) is defined as the linear program

$$\begin{aligned} \text{minimize} \quad & yb \\ \text{subject to} \quad & yA \geqslant c \\ & y \in \mathbb{R}^M_+. \end{aligned} \tag{1.2}$$

Feasible solutions to the dual problem provide upper bounds on the objective function value of the primal problem. This is known as *weak duality*. The next theorem is known as the *strong duality* theorem.

Theorem 1.4. *If the primal* (1.1) *has an optimal solution then the dual* (1.2) *has an optimal solution and their objective function values are equal.*

Given a polyhedron $P \subseteq \mathbb{R}^E$, the *separation problem* associated with P can be stated as follows. Given a point $y \in \mathbb{R}^E$, decide whether y belongs to P, and if not, find a vector $c \in \mathbb{R}^E$ such that $cx < cy$ for all x in P. When we are interested in algorithmic aspects, it is reasonable to assume that P is a *rational* polyhedron, i. e., we assume that there exist a matrix A and a vector b whose entries are rationals such that $P = \{x : Ax \leqslant b\}$. A polyhedron P is *integral* if every nonempty face of P contains an integral point. A $0, \pm 1$ matrix A is *totally unimodular* if for each integral vector b the polyhedron $\{x : x \geqslant 0,\ Ax \leqslant b\}$ is integral.

1.1.2 Supermodularity

In this subsection, we briefly review the concepts of supermodular systems and polyhedra. For a more extensive discussion as well as for an overview on the development of this concepts we refer the reader to the surveys of Lovász [Lov83] and Fujishige [Fuj84], and to Fujishige's monograph [Fuj91]. Using polyhedra associated with supermodular systems unifies and simplifies the study of several classes of scheduling polyhedra. The nature of these scheduling polyhedra becomes apparent in this supermodularity framework.

Let N be a finite set, and let $f : 2^N \to \mathbb{R}$ be a set–function that is *normalized*,

$$f(\emptyset) = 0$$

and *supermodular*,

$$f(S \cup T) + f(S \cap T) \geqslant f(S) + f(T) \ , \qquad \text{for all } S, T \subseteq N.$$

The latter inequality is equivalent to the *local supermodularity condition,*

$$f(S \cup \{j,k\}) - f(S \cup \{k\}) \geqslant f(S \cup \{j\}) - f(S)$$

for all $j,k \in N$, $j \neq k$, and $S \subseteq N \setminus \{j,k\}$.

The pair $(2^N, f)$ is called a *supermodular system*. If f is *submodular* instead of supermodular, i. e., if $-f$ is supermodular, then we obtain a *submodular system*. The *supermodular polyhedron* associated with a supermodular system is defined by

$$P(f) := \{x \in \mathbb{R}^N : x(S) \geqslant f(S) \text{ for all } S \subseteq N\} \ ,$$

where, as usual, for $x \in \mathbb{R}^N$ and $S \subseteq N$, we let $x(S) := \sum_{j \in S} x_j$. The corresponding *base polytope* $B(f)$ is the set of all points in the supermodular polyhedron that are minimal with respect to the usual partial order on $\mathbb{R}^N$; equivalently,

$$B(f) := \{x \in P(f) : x(N) = f(N)\} \ .$$

Notice that $P(f)$ is the *dominant* of $B(f)$, i. e., $P(f) = B(f) + \mathbb{R}^N_+$.

In the same way we can define a submodular polyhedron and its associated base polytope by reversing the sign of the inequalities. If $(2^N, f)$ is a supermodular system, and if $f^\sharp : 2^N \to \mathbb{R}$ is defined by $f^\sharp(S) := f(N) - f(N \setminus S)$, then $(2^N, f^\sharp)$ is a submodular system, which is called the *dual* system. Notice that this name is justified by observing that $(f^\sharp)^\sharp = f$. Furthermore, the corresponding base polytopes are identical, $B(f) = B(f^\sharp)$. In particular, a point $x \in \mathbb{R}^N$ with $x(N) = f(N)$ satisfies $x(S) \geqslant f(S)$ if and only if it satisfies $x(N \setminus S) \leqslant f^\sharp(N \setminus S)$.

Whereas a supermodular polyhedron is obviously full–dimensional, the dimension of its base polytope depends on f. A subset $S \subseteq N$ is *f–separable* if there exists a non–trivial partition $\{S_1, S_2\}$ of S with $f(S) = f(S_1) + f(S_2)$. Otherwise, S is called *f–inseparable*. The ground set N is generally f–inseparable for the functions f that arise from scheduling applications. Therefore, we will concentrate on this case. Under this assumption Shapley [Sha71] showed that the base polytope $B(f)$ is of dimension $|N| - 1$. An inequality $x(S) \geqslant f(S)$ defines a facet of $P(f)$ if and only if $S \neq \emptyset$ is f–inseparable. If N is f–inseparable, $x(S) \geqslant f(S)$ defines a facet of $B(f)$ if and only if $\emptyset \subset S \subset N$, S is f–inseparable, and $N \setminus S$ is $f^\sharp$–inseparable [Sch96a].

A supermodular function f is called *strictly supermodular* if it satisfies $f(S \cup T) + f(S \cap T) > f(S) + f(T)$ for all subsets S, T for which neither $S \subseteq T$ nor $S \supseteq T$ holds. Analogously, one can define strict submodularity. A set–function f is strictly supermodular if and only if its dual function $f^\sharp$ is strictly submodular. Observe that every subset of N is f–inseparable if f is strictly supermodular. Therefore, we need all the inequalities $x(S) \geqslant f(S)$ to describe the supermodular polyhedron and the base polytope of a strictly supermodular system.

In spite of the huge number of constraints defining supermodular polyhedra and base polytopes, there exists a simple procedure for optimizing a linear objective function over them. Assume that we want to minimize wx. If there exists an element j with $w_j < 0$ then the minimum over $P(f)$ is unbounded. Otherwise $w \geqslant 0$ and there exists an optimal solution that belongs to $B(f)$. To minimize over $B(f)$, assume that $N = \{1,\dots,n\}$ and $w_1 \geqslant w_2 \geqslant \cdots \geqslant w_n$. By proceeding greedily, i. e., defining x^* by

$$\begin{aligned}
x_1^* &:= \min\{x_1 : x \in B(f)\},\\
x_2^* &:= \min\{x_2 : x \in B(f), x_1 = x_1^*\},\\
&\vdots\\
x_n^* &:= \min\{x_n : x \in B(f), x_1 = x_1^*, x_2 = x_2^*, \dots, x_{n-1} = x_{n-1}^*\},
\end{aligned}$$

we obtain an optimal solution that can also be written as

$$\begin{aligned}
x_1^* &= f(\{1\}),\\
x_j^* &= f(\{1,\dots,j\}) - f(\{1,\dots,j-1\}), \quad j = 2,\dots,n.
\end{aligned}$$

The running time of this *greedy algorithm* is dominated by the sorting step, and it is therefore $O(n \log n)$. Here, we assume that the time needed for evaluating f is $O(1)$.

2 Natural Date Variables

In many nonpreemptive machine scheduling problems a schedule is uniquely determined by the corresponding job completion times C_j (or it can easily be (re)constructed given the job completion times). Furthermore, most performance measures involve the job completion times. Therefore, it is natural to characterize schedules by their completion time vectors and to study their convex hull. In this section, we take this point of view and summarize results obtained in this direction.

2.1 Disjunctive Constraints

We start our discussion of the convex hull of completion time vectors associated with feasible schedules with a quite general single machine scheduling problem that will be restricted to some "easier" cases in the following subsections. Assume we are given a single machine that is disjunctive, i. e., it can process at most one job at a time. Let $N = \{1,\dots,n\}$ be the set of jobs that have to be nonpreemptively processed on that machine, that is, once the processing of job j has started j cannot be unloaded during its processing time $p_j > 0$. If job j is the immediate predecessor of job k in a feasible schedule, we have to take into account a certain setup or changeover time $s_{jk} \geqslant 0$, e. g., for cleaning the machine, changing tools or fixtures, etc. We also allow for an initial changeover time s_{0j} which can be seen as a (job dependent) release date for the first job in the schedule. Furthermore, each feasible sequence has to be consistent with precedence constraints imposed by a given partial order $D = (N, A)$ on the set of jobs. That is, if $(j,k) \in A$, j has to be processed before job k in any feasible schedule. (Such precedence relations may occur because of technological reasons or may result from some dominance properties.) Every feasible schedule C is associated with a unique permutation $\pi : N \to \{1,\dots,n\}$ such that $C_{\pi^{-1}(1)} < C_{\pi^{-1}(2)} < \cdots < C_{\pi^{-1}(n)}$. In this way we derive only those permutations which extend the given partial order D. On the other hand, we can associate a so–called *permutation schedule* C^π with each linear extension π of D, namely,

$$\begin{aligned}
C^\pi_{\pi^{-1}(1)} &:= s_{0,\pi^{-1}(1)} + p_{\pi^{-1}(1)} && \text{and}\\
C^\pi_{\pi^{-1}(k)} &:= C^\pi_{\pi^{-1}(k-1)} + s_{\pi^{-1}(k-1),\pi^{-1}(k)} + p_{\pi^{-1}(k)} && \text{for } k = 2,\dots,n.
\end{aligned}$$

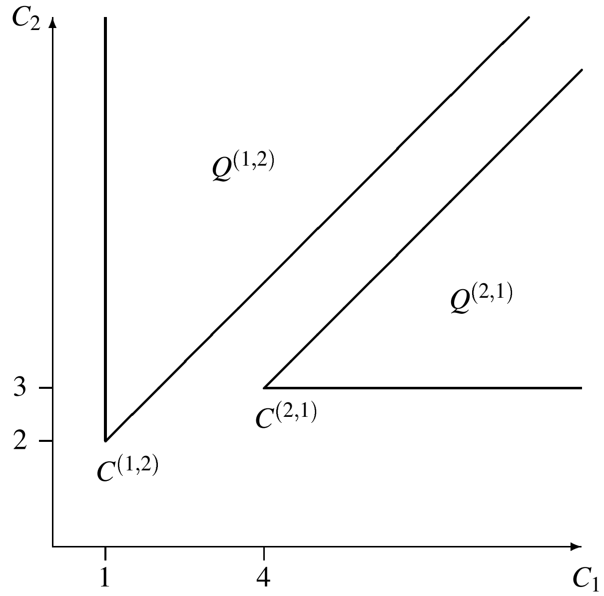

Figure 1: Disjunctive set of feasible schedules.

The permutation schedule of π is the only schedule associated with π without inserted idle time before any of the jobs. Moreover, the set Q of all feasible schedules is the union of the (translated) polyhedral cones

$$Q^{\pi} := \{C \in \mathbb{R}^N : C_{\pi^{-1}(k)} \geqslant C_{\pi^{-1}(k-1)} + s_{\pi^{-1}(k-1),\pi^{-1}(k)} + p_{\pi^{-1}(k)}, \quad k = 1,\ldots,n\} \ ,$$

one for each linear extension π of D (we assume that $C^{\pi}_{\pi^{-1}(0)} = 0$). Note that cone Q^{π} has as apex the schedule C^{π} and its extreme rays have directions $r^1,\ldots,r^n$, respectively, where $r^j := \sum_{\ell=j}^{n} e^{\pi^{-1}(\ell)}$ and e^k is the k–th unit vector in $\mathbb{R}^N$. Figure 1 shows the cones associated with a two–dimensional example, where $p_1 = p_2 = 1$, $s_{01} = s_{12} = s_{21} = 0$, and $s_{02} = 2$.

The convex hull of Q, which is the object we are interested in, is not necessarily a polyhedron, even though it is the convex hull of the union of a finite set of polyhedral cones. Observe that already in our example above $\text{conv}(Q)$ is not closed (see Queyranne and Wang [QW92] for details as well as for counterexamples from other combinatorial optimization problems). It follows from a result of Balas [Bal85] (see also [QW92]) that $\text{conv}(Q)$ is closed if the *triangle inequalities*

$$s_{jk} + p_k + s_{k\ell} \geqslant s_{j\ell} \qquad \text{for } j \in \{0,\ldots,n\}, k,\ell \in \{1,\ldots,n\}$$

are satisfied. Although this condition seems to be reasonable in many applications, Carter, Magazine and Moon [CMM88] pointed out situations where this condition is violated. Notice that the triangle inequalities are trivially satisfied, for example, if the changeover times s_{jk} are small, say, no larger than $\min_\ell p_\ell$, for all $j = 0,\ldots,n$ and all $k \neq j$.

We denote by $P_n(D)$ the convex hull of feasible schedules if there are no changeover times, and by $P_{n,s}(D)$ the one in the presence of changeover times, where we always assume that the triangle condition holds. If there are no precedence constraints, we simply write P_n and $P_{n,s}$ instead of $P_n(D)$ and $P_{n,s}(D)$, respectively.

2.2 Base Polytope vs. Dominant

In the absence of precedence constraints the recession cone of $P_{n,s}$ coincides with the nonnegative orthant. It follows that $P_{n,s}$ is of full dimension. This remains true when there is a partial order on the set of jobs, independent of its structure:

Proposition 2.1. *Let D be a precedence relation on the set $N = \{1,\dots,n\}$ of jobs. Then the scheduling polyhedron $P_{n,s}(D)$ is full–dimensional, i. e., $\dim(P_{n,s}(D)) = n$.*

Proof. Let π be a linear extension of D. Construct a feasible completion time vector C as follows:

$$C_j := \begin{cases} s_{0j} + p_j \ , & \text{if } j = \pi^{-1}(1), \\ C_{\pi^{-1}(k-1)} + 1 + s_{\pi^{-1}(k-1),j} + p_j \ , & \text{if } j = \pi^{-1}(k), k \in \{2,\dots,n\}, \end{cases} \qquad j \in N.$$

The $n+1$ points C and $C + e^j$, $j \in N$ belong to Q and are affinely independent. □

From now on we simplify the underlying scheduling problem by assuming that there are no changeover times. We will return to the problem with changeover times in Section 2.6.

Since we also do not allow for different job release dates, we can restrict to schedules without inserted idle times for optimizing a regular performance measure (cf., [CMM67]). As mentioned above, the permutation schedules are the only ones satisfying this additional condition. We denote the convex hull of all feasible permutation schedules by $B_n(D)$. Indeed, when there are no precedence constraints this polytope is the linear transformation of a base polytope of a supermodular system (see Section 2.3). We call it B_n in this case. $P_n(D)$ is a kind of dominant of $B_n(D)$ and it is indeed its dominant if the jobs are pairwise incomparable. The polytope $B_n(D)$ is also known under the name *generalized permutahedron* of the poset D (cf., [AS93, AS94]) since when all job processing times are equal to one, it is the convex hull of all permutations that extend D. The *permutahedron* (or *permutohedron* or *taxihedron*) is the convex hull of all permutations. It has almost independently been studied by Schoute [Sch11], Rado [Rad52] (see also [YKK84]), Guilbaud and Rosenstiehl [GR71], Benzécri [Ben71], Kreweras [Kre71], Balas [Bal75], Gaiha and Gupta [GG77], and Maes and Kappen [MK92]. The permutahedron of a poset has been the subject of investigation by von Arnim, Faigle and Schrader [AFS90], Schulz [Sch95], and von Arnim and Schrader [AS93].

In case that all job processing times are equal to one, all the work just mentioned is concerned with the polytope $B_n(D)$ rather than the polyhedron $P_n(D)$. Von Arnim and Schulz [AS94] also took this point of view and studied $B_n(D)$ whereas Queyranne [Que93] as well as Queyranne and Wang [QW91b] concentrate on $P_n(D)$. What are the respective advantages of using each class of polyhedra? $P_n(D)$ shares one property with many other polyhedra arising from combinatorial optimization problems, which makes it easier to study: it is full–dimensional. It follows in particular that the facet defining inequalities are unique up to positive multiples. Another advantage caused by its structure is that for any valid inequality $\sum_{j\in N} a_j C_j \geqslant \beta$ the sum of the coefficients, $\sum_{j\in N} a_j$ is always greater than or equal to zero. Otherwise we could take a schedule and by introducing sufficiently large idle time before the first job we would obtain a contradiction to the validity. Consequently, we may partition the class of valid inequalities in *positive–sum* and *zero–sum* inequalities, respectively. Observe further that a list of all facet inducing inequalities for $P_n(D)$ contains an inducing inequality for each facet of $B_n(D)$. Also, each inequality valid for $P_n(D)$ is valid for $B_n(D)$, but not vice versa. Another advantage of $P_n(D)$ that is again due to the structure of its recession cone is that all valid inequalities for $P_n(D)$ remain valid when nonnegative job release dates are introduced. This is especially important if the single machine problem is part of a more complicated problem involving multiple machines. If we minimize the weighted sum of completion times, $\sum_j w_j C_j$, an objective function which is obviously linear over $P_n(D)$, if there are no release dates different from zero and if all the weights are nonnegative, then we may always obtain an optimal solution that is an extreme point of $B_n(D)$.

On the other hand, the polytope $B_n(D)$ may contain more problem specific structure than $P_n(D)$. We will illustrate this. A poset $D = (N,A)$ is *series decomposable* in $D_1 = (N_1,A_1)$ and $D_2 = (N_2,A_2)$ if $N = N_1 \cup N_2$ with $N_1, N_2 \neq \emptyset$, $N_1 \cap N_2 = \emptyset$, and $(j,k) \in A$ for all $j \in N_1$ and all $k \in N_2$. We write $D = D_1 * D_2$ if D admits such a decomposition. The arc sets A_1 and A_2 are those subsets of A induced by N_1 and N_2, respectively. Note that every poset D has a unique *series decomposition* $D = D_1 * D_2 * \cdots * D_q$ where the nonempty suborders $D_1, D_2, \ldots, D_q$ are not further series decomposable. If the precedence relation D of the scheduling problem decomposes as $D = D_1 * \cdots * D_q$, it is enough to solve q smaller problems on $D_1, \ldots, D_q$, respectively, and to concatenate the partial solutions to obtain an entire optimal solution. This property carries over to $B_n(D)$ as we will show now.

Theorem 2.2. [Sch96a] *Let D be a poset with series decomposition $D_1 * \cdots * D_q$. Then $B_n(D)$ is the Cartesian product of the polytopes $B'_{n_1}(D_1), \ldots, B'_{n_q}(D_q)$ where $B'_{n_i}(D_i)$ arises from $B_{n_i}(D_i)$ through translation by $p(D_1 \cup \cdots \cup D_{i-1})\mathbb{1}$, i. e.,*

$$B_n(D) = B'_{n_1}(D_1) \times \cdots \times B'_{n_q}(D_q) \ .$$

Here, n_i denotes the number of jobs in $D_i = (N_i, A_i)$, i. e., $n_i := |N_i|$, for $i = 1, \ldots, q$.

Since a minimal description in terms of linear equations and inequalities for the Cartesian product of given polyhedra can be obtained by the juxtaposition of minimal linear systems of the given polyhedra, one may concentrate on posets that are not series decomposable, when studying $B_n(D)$. If D is not series decomposable, the dimension of $B_n(D)$ is $n-1$ [Sch96a]. With the help of Theorem 2.2 it is easy to determine the dimension of $B_n(D)$ for arbitrary D.

Corollary 2.3. [Sch96a] *Let D be a poset with series decomposition $D_1 * \cdots * D_q$. Then*

$$\dim(B_n(D)) = n - q \ .$$

Since all valid inequalities $\sum_{j\in N} a_j C_j \geqslant \beta$ for $P_n(D)$ are either positive–sum or zero–sum, it follows that any completion time vector induced by a schedule with inserted idle time is contained only in unbounded faces of $P_n(D)$. Thus, $B_n(D)$ is precisely the unique bounded face of $P_n(D)$ of maximal dimension, and all bounded faces of $P_n(D)$ are contained in $B_n(D)$.

2.3 Linear Description and Supermodularity

In this section, we assume that there are no precedence constraints. In this case a classical result of scheduling theory tells us how to minimize $\sum_j w_j C_j$, the weighted sum of completion times. According to Smith's rule [Smi56] we have to sequence the jobs in nonincreasing order of their ratios w_j/p_j. If we assume that $1, 2, \ldots, n$ is such an order, this leads to the optimal objective function value $\sum_{j=1}^n w_j \sum_{k=1}^j p_k$. Now, let the weights of the jobs be identical to their processing times. Then all the jobs have the same ratio and the optimal value does not depend on the order of the jobs. In fact, it becomes

$$\sum_{j=1}^{n} p_j \sum_{k=1}^{j} p_k = \frac{1}{2}\Big(\big(\sum_{j\in N} p_j\big)^2 + \sum_{j\in N} p_j^2\Big) \ .$$

For any subset S of jobs, let $p^2(S)$ (not to be confused with $p(S)^2 = p(S) \cdot p(S)$) denote $\sum_{j\in S} p_j^2$, and let

$$f(S) := \frac{1}{2}\left(p(S)^2 + p^2(S)\right) \ . \tag{2.1}$$

Then each feasible schedule C satisfies

$$\sum_{j\in S} p_j C_j \geqslant f(S) \quad \text{for all } S \subseteq N, \tag{2.2}$$

and the permutation schedules also satisfy

$$\sum_{j\in N} p_j C_j = f(N) \ . \tag{2.3}$$

After deriving this first class (2.2) of valid inequalities for P_n the question naturally arises whether these inequalities are enough. This is in fact the case and we now provide a different proof from that of Theorem 3.1 in [Que93]. First, observe that $f(\emptyset) = 0$ and

$$f(S\cup T) + f(S\cap T) = f(S) + f(T) + p(S\setminus T)p(T\setminus S), \quad \text{for all } S,T \subseteq N. \tag{2.4}$$

Since we assumed the processing times to be positive, (2.4) implies that the set–function $f : 2^N \to \mathbb{R}$ is strictly supermodular. Let $\overline{B}_n$ be the polytope defined by inequalities (2.2) and equation (2.3). By the variable transformation

$$\overline{C}_j := p_j C_j \qquad \text{for } j = 1,\dots,n, \tag{2.5}$$

we see that $\overline{B}_n$ is a linear transformation of the base polytope $B(f)$ of the strictly supermodular system on N defined by f.

Theorem 2.4. *The scheduling polytope B_n is completely described by inequalities* (2.2) *and equation* (2.3)*, i. e., $B_n = \overline{B}_n$.*

Proof. Since we know about the validity of the inequalities in question, we only have to show that $\overline{B}_n \subseteq B_n$. Let C be an arbitrary vertex of $\overline{B}_n$, and let $\overline{C}$ be the vertex of $B(f)$ assigned to C by (2.5). Furthermore, let w be a vector such that $\overline{C}$ is the unique minimum of the linear programming problem $\min\{wx : x \in B(f)\}$. We may assume (by renumbering of jobs) that $w_1 \geqslant w_2 \geqslant \cdots \geqslant w_n$. Then the greedy algorithm for supermodular polyhedra implies, that

$$\begin{aligned} \overline{C}_j &= f(\{1,\dots,j\}) - f(\{1,\dots,j-1\}) \\ &= p_j^2 + p_j\Big(\sum_{k=1}^{j-1} p_k\Big), \quad \text{for } j = 1,\dots,n, \end{aligned}$$

where we used the definition of the set–function f to obtain the last equation. Thus, by the reverse linear transformation it follows that

$$C_j = \sum_{k=1}^{j} p_k \quad \text{for } j = 1,\dots,n.$$

We conclude that C is the completion time vector of the feasible sequence $1,\dots,n$. □

Theorem 2.4 not only provides a complete description of the scheduling polytope B_n in terms of linear equations and inequalities, it also implies that B_n as well as P_n are linear transformations of the base polytope $B(f)$ and the supermodular polyhedron $P(f)$, respectively, where f is defined by (2.1). Since f is strictly supermodular, we have the following corollary.

Corollary 2.5.

(a) *For each nonempty subset $S \subseteq N$ the inequality $\sum_{j\in S} p_jC_j \geqslant f(S)$ induces a facet of the scheduling polyhedron P_n. These are all facet defining inequalities of P_n (up to positive multiples).*

(b) *For each nonempty and proper subset $S \subset N$ the inequality $\sum_{j\in S} p_jC_j \geqslant f(S)$ induces a facet of the scheduling polytope B_n.*

Using the linear transformation (2.5), minimizing $\sum_{j\in N} w_jC_j$ over B_n is equivalent to minimizing $\sum_{j\in N}(w_j/p_j)\overline{C}_j$ over $B(f)$. But the greedy algorithm for supermodular polyhedra solves this problem: sort the jobs in nonincreasing order of the ratios w_j/p_j. The proof of Theorem 2.4 implies that it is optimal to process the jobs in this order. Thus, Smith's rule is a special instance of the greedy algorithm for supermodular polyhedra, as observed first by Queyranne [Que93].

The dual function $f^\sharp$ of f implies "another" set of inequalities that are sufficient to describe B_n together with equation (2.3), namely

$$\sum_{j\in S} p_jC_j \leqslant f^\sharp(S) \quad \text{for all } S \subset N, \tag{2.6}$$

where

$$f^\sharp(S) = \frac{1}{2}\left(p(S)^2 + p^2(S)\right) + p(S)p(N \setminus S) \ . \tag{2.7}$$

The set of all faces of a polyhedron P forms under set inclusion a finite lattice (cf., e. g., [Zie95]). Queyranne [Que93] observed that the face lattice of P_n is isomorphic to the lattice of all ordered subpartitions of N under refinement. Furthermore, it is well–known that the face lattice of the permutahedron is isomorphic to the lattice of all ordered partitions of N. Schulz [Sch93a] pointed out that the face lattices of the base polytopes of all strictly supermodular systems are actually isomorphic to the lattice of all ordered partitions. The same holds for strictly supermodular polyhedra and the set of ordered subpartitions. By the linear transformation (2.5) the face lattices of the scheduling polyhedra B_n and P_n are also isomorphic to these lattices, respectively, and one can derive nice formulae for the number of faces of various dimensions and conditions on adjacency relationships (see [Que93] and [Sch93a]).

2.4 Precedence Constraints

We now come back to the single machine scheduling problem with precedence constraints. We are given a poset D on the set of jobs and we are interested in linear descriptions of $B_n(D)$ and $P_n(D)$, respectively. Since the presence of precedence constraints makes the problem to minimize the weighted sum of completion times, $1|\text{prec}|\sum w_jC_j$, NP–hard – even if all weights or all processing times are equal to one (see [Law78] or [LRK78]) – we cannot in general hope for finding a complete linear description. But some authors derived partial descriptions that turn out to be sufficient when the poset D has a special structure.

Let $D_1 = (N_1, A_1)$ and $D_2 = (N_2, A_2)$ be posets on disjoint ground sets N_1 and N_2, respectively. The *parallel composition* $D_1 + D_2 = (N, A_+)$ and the *series composition* $D_1 * D_2 = (N, A_*)$ of D_1 and D_2 are posets on $N := N_1 \cup N_2$ defined by

$$(j,k) \in A_+, \text{ if } \begin{cases} (j,k) \in A_1 \quad \text{and} \quad j,k \in N_1 \quad \text{or} \\ (j,k) \in A_2 \quad \text{and} \quad j,k \in N_2 \end{cases}$$

and

$$(j,k) \in A_*, \text{ if } \begin{cases} (j,k) \in A_+ \quad \text{or} \\ j \in N_1 \quad \text{and} \quad k \in N_2. \end{cases}$$

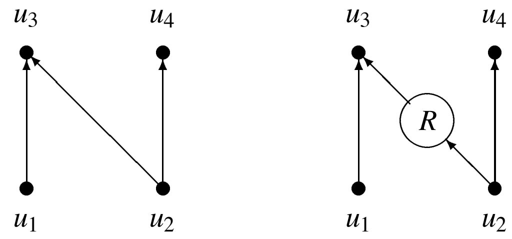

Figure 2: The order $\mathbf{N}$ and a spider $\mathbf{N} \bowtie R$.

Let $\mathbf{N}$ denote the 4–element order shown in Figure 2. If (R, A_R) is another poset disjoint from $\mathbf{N}$, then the *spider composition* $\mathbf{N} \bowtie R$ of $\mathbf{N}$ and R is the poset $(\mathbf{N} \cup R, A)$ defined on $\mathbf{N} \cup R$ by

$$(j,k) \in A, \text{ if } \begin{cases} (j,k) \in A_{\mathbf{N}} & \text{and} \quad j,k \in \mathbf{N}, \quad \text{or} \\ (j,k) \in A_R & \text{and} \quad j,k \in R, \quad \text{or} \\ j = u_2 & \text{and} \quad k \in R, \quad \text{or} \\ k = u_3 & \text{and} \quad j \in R. \end{cases}$$

A *weak* poset is a poset that is defined as the series composition of antichains. (A poset is an antichain if all its elements are pairwise incomparable.) A poset is *series–parallel* if it is either a singleton or the parallel or series composition of two series–parallel posets. A poset is N–*sparse* if it is a singleton, if it is the parallel or series composition of two N–sparse posets, or if it is a spider composition $\mathbf{N} \bowtie R$ where R is either the empty set or N–sparse. Obviously, each weak poset is series–parallel and each series–parallel poset is N–sparse, but not vice versa. Notice that we can associate naturally a decomposition (or parse) tree with each N–sparse poset.

Assuming the decomposition tree is given, Lawler [Law78] proposed an $O(n \log n)$ algorithm for the sequencing problem when the precedence constraints are series–parallel. This algorithm starts at the bottom of the decomposition tree and works upward, finding an optimal sequence for a node from previously determined optimal sequences for its children. Schulz [Sch93b] extended this algorithm to N–sparse precedence constraints by also handling spider compositions. His algorithm has running time $O(n^2)$.

2.4.1 Parallel Inequalities

Since $B_n(D) \subseteq B_n(D')$ if $D = (N, A)$ is a poset extending $D' = (N, A')$, i. e., if $A' \subseteq A$, the inequalities we derived for B_n remain valid for $B_n(D)$. However, it is easy to see that an inequality

$$\sum_{j \in I} p_j C_j \geqslant f(I) \tag{2.8}$$

induces a nonempty face of $B_n(D)$ and $P_n(D)$ if and only if I is an *ideal* (or *initial set*) of D. A subset $I \subseteq N$ is an ideal of the poset D if for every $k \in I$ and every j with $(j,k) \in A$ we have $j \in I$. For an ideal I the inequality (2.8) is therefore called an *ideal constraint* or a *parallel inequality*. It follows from Theorems 2.2 and 2.4 that ideal inequalities are essentially sufficient for defining the scheduling polytope $B_n(D)$ when D is a weak order.

Corollary 2.6. [Sch93a] *Let D be a poset with series decomposition $D_1 * D_2 * \cdots * D_q$, where the components D_i are antichains. Then*

$$\begin{aligned} \sum_{j\in D_i} p_j C_j &= f(D_i) + p(D_i)p(D_1 \cup \cdots \cup D_{i-1}) && \text{for } i = 1,\ldots,q, \\ \sum_{j\in D_1\cup\cdots\cup D_{i-1}\cup I} p_j C_j &\geqslant f(D_1 \cup \cdots \cup D_{i-1} \cup I) && \text{for } \emptyset \subset I \subset D_i,\ i = 1,\ldots,q, \end{aligned}$$

is a complete and minimal linear system describing $B_n(D)$.

Queyranne and Wang [QW91b] proved that all parallel inequalities induced by non series decomposable ideals of the job set N define facets of the polyhedron $P_n(D)$. They also show that these are, up to positive multiples, the only positive–sum facet inducing inequalities for $P_n(D)$. If, in particular, D is itself non series decomposable, it turns out that not all these inequalities remain facet defining for $B_n(D)$. In fact, since the face of $B_n(D)$ induced by the ideal $I \subset N$ coincides with $B_n(I * (N \setminus I))$, Corollary 2.3 implies that I induces a facet of $B_n(D)$ if and only if both I and $N \setminus I$ are not series decomposable. By applying Theorem 2.2 we obtain the following theorem due to Schulz [Sch93a, Sch95] for arbitrary D.

Theorem 2.7. *Let D be a poset with series decomposition $D_1 * \cdots * D_q$. An ideal $I = D_1 * \cdots * D_\ell * \hat{I}$, $\ell \in \{0,\ldots,q-1\}$, defines a facet of $B_n(D)$ if and only if $\emptyset \subset \hat{I} \subset D_{\ell+1}$ and both $\hat{I}$ and $D_{\ell+1} \setminus \hat{I}$ are not series decomposable.*

If we assume that in the initial LP relaxation of a cutting plane algorithm for $1|\text{prec}|\sum w_j C_j$ the simple precedence inequalities

$$C_k \geqslant C_j + p_k \qquad \text{for all } (j,k) \in A \tag{2.9}$$

are included, the separation algorithm of Queyranne [Que93] that is based on the following lemma determines an ideal whose associated inequality is violated, if one exists.

Lemma 2.8. [Que93] *Let C be a point in $\mathbb{R}^N$. If $I \subseteq N$ maximizes $f(I) - \sum_{j\in I} p_j C_j$, then $k \in I$ if and only if $C_k \leqslant p(I)$.*

Lemma 2.8 implies that if the jobs are sorted in nondecreasing order of the C_j's, say, $C_1 \leqslant C_2 \leqslant \cdots \leqslant C_n$, then an optimum subset I can be found among the n sets $\{1,\ldots,k\}$, $k = 1,\ldots,n$. This observation leads immediately to an $\mathrm{O}(n \log n)$ separation algorithm (see [Que93] and [QW91a] for details).

2.4.2 Error Bounds on the Optimality Gap

We now consider the quality of the lower bound obtained from the LP formulation including all the precedence constraints (2.9) and, through the $\mathrm{O}(n \log n)$ separation algorithm just mentioned, all the parallel inequalities (2.2). For this, we also consider the quality of certain heuristics (approximation methods) which construct feasible schedules. In the case of precedence constraints, the following simple idea [QW91a] produces a feasible schedule: given any vector C satisfying the following relaxation of the precedence constraints (2.9)

$$C_k > C_j \quad \text{for all } (j,k) \in A, \tag{2.10}$$

sequence the jobs in the order of the components of C. Although condition (2.10) is sufficient to guarantee feasibility of the resulting schedule, we need additional conditions on C for this schedule to be near–optimal.

Assume now that C satisfies all the parallel inequalities (2.2). Let C^H denote the completion time vector of the schedule obtained by sequencing the jobs in the order of the components of C. To simplify the notation, assume that this order is $1,2,\ldots,n$; that is, we had $C_1 \leqslant C_2 \leqslant \ldots \leqslant C_n$. Note that, for a single machine and in the absence of release dates and changeover times,

$$C_j^H = \sum_{k=1}^{j} p_k \quad \text{for all } j = 1,\ldots,n. \tag{2.11}$$

The following lemma, due to Schulz [Sch96b] (see also Hall et al. [HSSW96]), implies the *job-by-job bound:*

$$C_j^H < 2C_j \qquad \text{for every } j = 1,\ldots,n. \tag{2.12}$$

Lemma 2.9. [Sch96b] *Assume that vector $C \in \mathbb{R}^N$ satisfies $C_1 \leqslant C_2 \leqslant \cdots \leqslant C_n$ and all the parallel inequalities* (2.2). *Then*

$$C_j > \frac{1}{2}\sum_{k=1}^{j} p_k \qquad \textit{for all } j = 1,\ldots,n. \tag{2.13}$$

Proof. For any $j \in N$, let $S = \{1,\ldots,j\}$ and $p(S) = \sum_{k=1}^{j} p_k$. Then the inequalities $C_j \geqslant C_k$ for all $k \in S$ and the parallel inequality (2.2) for S imply:

$$p(S)C_j = \big(\sum_{k\in S} p_k\big)C_j \geqslant \sum_{k\in S} p_k C_k \geqslant f(S) > \frac{1}{2}\big(\sum_{j\in S} p_j\big)^2 = \frac{1}{2}p(S)^2. \tag{2.14}$$

The result now follows from dividing by $p(S) > 0$. □

Theorem 2.10. [Sch96b] *Let C^{LP} denote an optimal solution to the LP relaxation*

$$\min \sum_{j\in N} w_j C_j \tag{2.15}$$

$$\text{subject to (2.9) and (2.2)} \tag{2.16}$$

of the scheduling problem $1|\text{prec}|\sum w_j C_j$ *(where all $w_j \geqslant 0$), and let C^H denote the completion time vector of the feasible schedule defined by sequencing the jobs in the same order as in C^{LP}. Then*

$$wC^{LP} > \frac{1}{2}wC^* \quad \textit{and} \quad wC^H < 2wC^* \tag{2.17}$$

where C^* denotes any optimal schedule.

Proof. As observed above, the fact that C^{LP} satisfies the inequalities (2.9) implies that the schedule C^H satisfies all precedence constraints. Assuming as above that $C_1^{LP} \leqslant C_2^{LP} \leqslant \ldots \leqslant C_n^{LP}$, we have, for all $j = 1,\ldots,n$, $C_j^H = \sum_{k=1}^{j} p_k$ and thus, from Lemma 2.9, the job-by-job bound $C_j^H < 2C_j^{LP}$. From $wC^{LP} \leqslant wC^* \leqslant wC^H$ and all $w_j \geqslant 0$ (with at least one $w_j > 0$), we obtain

$$wC^* \leqslant wC^H < 2wC^{LP} \leqslant 2wC^*.$$

The results follow. □

Thus the LP relaxation provides a lower bound which is more than 50% of the optimal value, and the corresponding heuristic schedule C^H has value less than twice the optimal value. Examples given in [HSSW96] show that both bounds are asymptotically tight.

Queyranne and Wang [QW91a] report computational experiments showing that the LP relaxation and LP-based heuristic perform much better in practice than the worst–case bounds in Theorem 2.10 above. They use a cutting plane algorithm to solve the LP relaxation (2.15)–(2.16), and supplement the LP-based heuristic with some simple job interchanges. The instances, with up to 160 jobs, were randomly generated with processing times and weights drawn from the discrete uniform distributions on $[1,100]$ and $[1,10]$, respectively. They find relative optimality gaps of less than 1% on *all* of the 260 instances generated. Queyranne and Wang also observe that, from Theorem 2.4, the LP bound from (2.15)–(2.16) is the optimal value of the Lagrangian relaxation considered by van de Velde [Vel95].

2.4.3 Series Inequalities

The simple precedence inequalities (2.9) may be generalized by replacing the jobs j and k by subsets of jobs. Let $J*K$ be a series decomposable subset of D. (To keep the notation simple, when we speak of subsets of the poset D we mean subsets of the job set N together with the precedence constraints induced by D.) That all the jobs in J have to precede all the jobs in K and that jobs may not overlap imply the following linear inequality. Let C be an arbitrary feasible schedule, and let t be the earliest time at which all jobs of J are completed, i. e., $t := \max\{C_j : j \in J\}$. Let $N_{\leqslant t}$ be the set of jobs completed no later than t, and $N_{>t}$ the set of jobs completed after t, so $J \subseteq N_{\leqslant t}$ and $K \subseteq N_{>t}$. Furthermore, let $D_{>t}$ be the poset on $N_{>t}$ induced by D, and $D_{\leqslant t}$ the inverse of the poset on $N_{\leqslant t}$ induced by D. Then, the appropriately shifted subvectors of C indexed by the jobs in $N_{>t}$ and $N_{\leqslant t}$ satisfy the ideal constraints for $P_{|N_{>t}|}(D_{>t})$ and $P_{|N_{\leqslant t}|}(D_{\leqslant t})$ associated with K and J, respectively:

$$\sum_{j\in K} p_j(C_j - t) \;\geqslant\; f(K) \;, \qquad (2.18)$$

$$\sum_{j\in J} p_j(t - C_j + p_j) \;\geqslant\; f(J) \;. \qquad (2.19)$$

Note that (2.19) is obtained by applying time reversal, starting at time t. By multiplying (2.18) with $p(J)$ and (2.19) with $p(K)$ and summing up the resulting inequalities, we obtain

$$\begin{aligned} & p(J)\sum_{j\in K} p_jC_j - p(K)\sum_{j\in J} p_jC_j \\ & \geqslant \frac{1}{2}p(J)p(K)\big(p(J)+p(K)\big) + \frac{1}{2}p(J)p^2(K) - \frac{1}{2}p(K)p^2(J) \;. \end{aligned} \qquad (2.20)$$

This inequality does not depend on t and therefore must be satisfied by every feasible schedule C. Hence, it is valid for $P_n(D)$. From its derivation it follows that it can only induce a nontrivial face if J is an ideal of $D_{\leqslant t}$ and K is an ideal of $D_{>t}$. That is, $J*K$ has to be an intermediate set. A set $S \subseteq N$ is called *intermediate* or *convex* (with respect to D) if $(j,k),(k,\ell) \in A$ and $j,\ell \in S$ imply $k \in S$. Inequality (2.20) which has been derived by von Arnim, Faigle and Schrader [AFS90] for the permutahedron and by Queyranne and Wang [QW91b] for $P_n(D)$ is called *series inequality* or *convex set constraint*. Queyranne and Wang showed that an intermediate set $J*K$ induces a facet of $P_n(D)$ if and only if neither J nor K is itself series decomposable. Necessity is easy to see: in case that J or K is series decomposable, say, $J = J_1 * J_2$, routine calculations show that the face induced by $J*K$ is the intersection of the faces induced by $J_1 * J_2$ and $J_2 * K$, respectively. Again, this condition is not sufficient in case of $B_n(D)$. Assume that D is non series decomposable. Then von Arnim and Schulz [AS94] (see also [Sch93a]) proved that an intermediate set $A*B$ induces a facet of $B_n(D)$ if and only if A, B, and $D/(A*B)$ are not series decomposable. Here $D/(A*B)$ stands for the contracted poset resulting from D by replacing $A*B$ by a single element.

While the parallel inequalities are sufficient to describe the scheduling polyhedra of weak posets, von Arnim, Faigle and Schrader [AFS90] and Queyranne and Wang [QW91b] showed for the permutahedron and for $P_n(D)$, respectively, that the parallel and the series inequalities are sufficient for series–parallel posets. Queyranne and Wang's proof exploits the decomposition tree and reveals similarities with Lawler's algorithm. We restate their main theorem to illustrate this. A *filter* (or *terminal set*) of a poset is a subset whose complement forms an ideal.

Theorem 2.11. [QW91b]

(a) *All facet inducing inequalities for* $P_n(D_1 + D_2)$ *are*

- (i) *all facet inducing inequalities for* $P_{|D_1|}(D_1)$,
- (ii) *all facet inducing inequalities for* $P_{|D_2|}(D_2)$,
- (iii) $\sum_{j \in I_1 \cup I_2} p_j C_j \geqslant f(I_1 \cup I_2)$, *where* I_1 *and* I_2 *are nonempty ideals of* D_1 *and* D_2, *respectively.*

(b) *All facet inducing inequalities for* $P_n(D_1 * D_2)$, *are*

- (i) *all facet inducing inequalities for* $P_{|D_1|}(D_1)$,
- (ii) *all zero–sum facet inducing inequalities for* $P_{|D_2|}(D_2)$,
- (iii)

$$p(F)\sum_{j\in I} p_jC_j - p(I)\sum_{j\in F} p_jC_j \geqslant \frac{1}{2}p(F)p(I)\big(p(F)+p(I)\big) + \frac{1}{2}p(F)p^2(I) - \frac{1}{2}p(I)p^2(F) \ ,$$

where F *is a non series decomposable filter of* D_1 *and* I *is a non series decomposable ideal of* D_2.

Minimal linear descriptions for $P_n(D)$ and $B_n(D)$ for series–parallel D are given in Queyranne and Wang [QW91b] and in Schulz [Sch93a], respectively.

The separation problem associated with the class of series inequalities is not completely solved. Queyranne and Wang [QW91a] proposed an algorithm for the *simple* series inequalities. This subclass contains only those series inequalities induced by $J * K$ where J or K is a singleton. Schulz [Sch93a] extended their algorithm to the case where J or K is of fixed size, say of size q. This leads to an $O(n^{q+1})$ algorithm. On the other hand, it follows from a result discussed in Section 4.2 below (see Theorem 4.4) that the separation problem for a class of inequalities that contains the series inequalities is solvable in polynomial time, but no combinatorial algorithm is known at the time of this writing.

In addition to the results and computational experiments mentioned previously, Queyranne and Wang [QW91a] show that the LP relaxation consisting of the parallel inequalities (2.2) and the simple simple series inequalities, for which they have fast separation algorithms, yields a lower bound never worse than that from the Lagrangian relaxation using slack variables proposed by Hoogeven and van de Velde [HV95b]. Margot, Queyranne and Wang [MQW96] extend the computational work of [QW91a] and develop a branch–and–cut algorithm for the same problem. They solve most instances of size up to 120 jobs with a modest number of nodes in the branching tree and within a few seconds of CPU time on a workstation. These empirical results suggest the question whether the bounds in (2.17) can be improved by the addition to the LP relaxation (2.15)–(2.16) of classes of valid inequalities such as simple series, general series, or the spider inequalities of the next section.

2.4.4 Spider Inequalities

Queyranne and Wang [QW91b] observed that as soon as the precedence constraints fail to be series–parallel the parallel and the series inequalities are no longer sufficient for describing the scheduling polyhedra. They showed that certain additional inequalities associated with the order **N** are needed. This is not surprising since the absence of an induced **N** is a characterization of series–parallel posets by a forbidden substructure. Von Arnim and Schrader [AS93] introduced a class of so–called *spider inequalities* that generalizes the **N**–inequalities of Queyranne and Wang. They showed that the addition of spider inequalities to all the inequalities we derived before leads to a complete linear description of $B_n(D)$ when D is N–sparse. The spider inequalities associated with a spider $S = \mathbf{N} \bowtie R$ (see Figure 2 above) can be stated as follows:

$$\begin{aligned} &p(S)\left[\Big(p(F)+p_{u_1}\Big)C_{u_3} - \sum_{j \in F \cup \{u_1\}} p_j C_j - \beta C_{u_2}\right] + \beta \sum_{j \in S} p_j C_j \\ &\geqslant p(S)\left[\frac{1}{2}\Big(p(F)+p_{u_1}\Big)\Big(p(F)+p_{u_1}+2p_{u_3}\Big) - \frac{1}{2}\Big(p^2(F)+p_{u_1}^2\Big) - \beta p_{u_2}\right] \\ &\quad + \frac{1}{2}\beta\Big[p(S)^2 + p^2(S)\Big], \end{aligned} \tag{2.21}$$

and

$$\begin{aligned} &p(S)\left[\sum_{j \in I \cup \{u_4\}} p_j C_j - \Big(p(I)+p_{u_4}\Big)C_{u_2} + \gamma C_{u_3}\right] - \gamma \sum_{j \in S} p_j C_j \\ &\geqslant p(S)\left[\frac{1}{2}(p(I)+p_{u_4})^2 + \frac{1}{2}(p^2(I)+p_{u_4}^2)\right] + \frac{1}{2}\gamma\Big[p(S)^2 - p^2(S)\Big], \end{aligned} \tag{2.22}$$

where $\beta = p(R \setminus F) + p_{u_2}$, $\gamma = p(R \setminus I) + p_{u_3}$, I is an ideal, and F is a filter of R.

Von Arnim and Schulz [AS94] show that a spider $S \subseteq D$ together with an ideal I or an filter F of R defines a facet of $B_n(D)$ if and only if S is an intermediate set and the contracted poset D/S is non series decomposable. They also present a minimal linear description that characterizes $B_n(D)$ completely when D is N–sparse.

2.5 Release Dates

We now consider single machine scheduling problems subject to release date constraints

$$C_j \geqslant r_j + p_j \quad \text{for all } j \in N, \tag{2.23}$$

where $r_j \geqslant 0$ is a given *release date* for job j. Note that all earlier results implicitly assumed a common release date $r_j = 0$ for all jobs.

The polyhedral structure of single–machine scheduling polyhedra subject to release dates is not well understood. See the next section for general results from Balas [Bal85] which apply to this special case of the problem considered therein. In the rest of this section, we consider recent approximation results based on simple (and rather weak) LP relaxations.

The simplest relaxation, studied by Schulz [Sch96b], uses only the release date inequalities (2.23) and, since all $r_j \geqslant 0$, the parallel inequalities (2.2).

Proposition 2.12. [Sch96b] *Let $C \in \mathbb{R}^N$ denote any vector satisfying all inequalities* (2.23) *and* (2.2). *As earlier and to simplify notation, assume that the jobs are numbered such that $C_1 \leqslant C_2 \leqslant \cdots \leqslant C_n$.*

Let C^H denote the completion time vector of the feasible schedule defined by sequencing the jobs in the same order as in C, that is,

$$C_1^H = r_1 + p_1 \qquad \textit{and} \qquad C_j^H = \max\{r_j, C_{j-1}^H\} + p_j \quad \textit{for } j = 2, \ldots, n. \tag{2.24}$$

Then, for all jobs $j = 1, \ldots, n$, we have

$$C_j^H < 3C_j. \tag{2.25}$$

Proof. Note that, for every job j, we have $C_j^H = r_k + \sum_{\ell=k}^{j} p_\ell$ for some job k scheduled before j (possibly job j itself), that is, with $C_k \leqslant C_j$. Letting $S = \{\ell \in N : k \leqslant \ell \leqslant j\}$, note that, as in the proof of Lemma 2.9, we have from $C_j \geqslant C_\ell$ for all $\ell \in S$ and from (2.2),

$$\big(\sum_{\ell \in S} p_\ell\big) C_j \geqslant \sum_{\ell \in S} p_\ell C_\ell \geqslant f(S) > \frac{1}{2}\big(\sum_{\ell \in S} p_\ell\big)^2,$$

implying $\sum_{\ell \in S} p_\ell < 2C_j$. The proposition then follows from this inequality and from $r_k < C_k \leqslant C_j$. □

Corollary 2.13. [Sch96b] *Let C^{LP} denote an optimal solution to the LP relaxation*

$$\min \sum_{j \in N} w_j C_j \tag{2.26}$$

$$\text{subject to (2.23) and (2.2)} \tag{2.27}$$

of the scheduling problem $1|r_j|\sum w_j C_j$ (where all $w_j \geqslant 0$), and let C^H denote the completion time vector of the feasible schedule defined by sequencing the jobs in the same order as in C^{LP}, as per equations (2.24)*. Then*

$$wC^{LP} > \frac{1}{3} wC^* \quad \textit{and} \quad wC^H < 3wC^* \tag{2.28}$$

where C^* denotes any optimal schedule.

Wang [Wan96] shows that the bounds in Corollary 2.13 are asymptotically tight. Note that, while scheduling the jobs according to the order in C^{LP}, a job j may fit in a "gap" (idle time interval) after r_j but before C_{j-1}^H. Thus, let $C^{H'}$ denote the feasible schedule whereby every job j is considered after job $j-1$ and then scheduled as early as possible. It is clear that $C_j^{H'} \leqslant C_j^H$ for all jobs j, so $wC^{H'} \leqslant wC^H < 3wC^*$, but this performance result cannot be improved in the worst case.

Schulz [Sch96b] also observes that the same bounds hold when we have precedence constraints in addition to release dates, provided of course that we add the simple precedence constraints (2.9) to the LP relaxation:

Corollary 2.14. [Sch96b] *The conclusions of Corollary 2.13 also apply to the LP relaxation*

$$\min \sum_{j \in N} w_j C_j \tag{2.29}$$

$$\text{subject to (2.23), (2.2) and (2.9)} \tag{2.30}$$

of the scheduling problem $1|r_j, \text{prec}|\sum w_j C_j$ (where all $w_j \geqslant 0$).

We now turn to a stronger relaxation for $1|r_j|\sum w_j C_j$. For any job subset $S \subseteq N$, let $r_{\min}(S) = \min_{j \in S} r_j$. Since no job in S can start before time $r_{\min}(S)$, we may think of this time as a new (provisional) time origin for job subset S. Therefore any feasible schedule C must satisfy

$$\sum_{j \in S} p_j \big(C_j - r_{\min}(S)\big) \geqslant f(S).$$

Defining $l(S) = (\sum_{j \in S} p_j) r_{\min}(S) + f(S)$, we see that every feasible schedule C must satisfy the *shifted parallel inequalities*

$$\sum_{j \in S} p_j C_j \geqslant l(S) \quad \text{for all } S \subseteq N. \tag{2.31}$$

Note that these shifted parallel inequalities imply both the release date constraints (2.23) (using $S = \{j\}$) and (since all $r_j \geqslant 0$) the parallel inequalities (2.2). Therefore the LP relaxation

$$\min \sum_{j \in N} w_j C_j \tag{2.32}$$

$$\text{subject to (2.31)} \tag{2.33}$$

to the scheduling problem $1|r_j|\sum w_j C_j$ yields a lower bound which is no smaller than that from the earlier LP relaxation (2.26)–(2.27).

(*Remark:* Note that the argument used in the proof of Proposition 2.12 cannot be used here to show $C_j^H < 2C_j$, because we may have $r_{\min}(\{k, \ldots, j\}) < r_k$; in fact, simple examples show that, using the LP relaxation (2.32)–(2.33), the uniform factor of 3 in the job-by-job bound (2.25) cannot be improved to 2 in the worst case.)

Goemans [Goe96] studies this shifted parallel inequalities relaxation in detail. Though the set function l is not supermodular in general, he shows that the polyhedron defined by all shifted parallel inequalities (2.31) is the affine image of a supermodular polyhedron; that is, the upper Dilworth truncation of l is a supermodular function. Using this property, Goemans shows that the LP relaxation (2.32)–(2.33) can be solved in $O(n \log n)$ time by a greedy algorithm. He also characterizes the facet defining shifted parallel inequalities and shows that the polyhedron defined by (2.31) is a projection of that defined by a "weak" time-indexed formulation from Dyer and Wolsey [DW90], see Subsection 3.5 below. This implies in particular that the optimal value of the LP relaxation (2.32)–(2.33) coincides with that of Dyer and Wolsey's time-indexed formulation.

In a subsequent paper [Goe97], Goemans describes a randomized algorithm for $1|r_j|\sum w_j C_j$ based on an optimal solution C^{LP} to the LP relaxation (2.32)–(2.33). Using an amortized approach (instead of the job-by-job approach above) and the properties of optimal solutions (instead of feasible solutions, as above) to the shifted parallel inequalities formulation (2.32)–(2.33) and to the LP relaxation of Dyer and Wolsey's time-indexed formulation, he shows that his randomized algorithm is a 2-approximation algorithm. In terms of the quality of the lower bounds, his results imply in particular that the shifted parallel inequalities lower bound is at least half the optimum value:

Theorem 2.15. [Goe97] *Let C^{SPI} denote an optimal solution to the shifted parallel inequalities LP relaxation* (2.32)–(2.33). *Then*

$$wC^{SPI} \geqslant \frac{1}{2} wC^* \tag{2.34}$$

where C^* denotes any optimal schedule.

Note that Theorem 2.15 also applies to the LP relaxation of Dyer and Wolsey's time-indexed formulation. Based on the same relaxations to $1|r_j|\sum w_j C_j$, but with an improved use of randomness Schulz and Skutella [SS] design a 1.847–approximation algorithm.

2.6 Sequence–Dependent Processing Times

In this final section on the natural date variables we reintroduce the single machine problem with changeover times (or setup times), but without precedence constraints, describing very briefly the work of Balas [Bal85]. By combining changeover times s_{jk} and job processing times p_k we obtain sequence–dependent processing times $p_{jk} := s_{jk} + p_k$, and this explains the title of this section. Balas started with a computationally very challenging problem, the *job shop problem*. In a job shop we are given a set of m disjunctive machines and each job j splits into (at most) m operations $j_i, i = 1, \dots, m$. Operation j_i has to be processed on machine i, and there is a specified processing order for the operations of each job. The well–known disjunctive graph model for this problem (cf., e. g., [Bal69]) translates into the following disjunctive formulation:

$$C_{j_i} \geqslant C_{j_h} + p_{j_h, j_i}, \qquad \text{if operation } h \text{ of job } j \text{ has to precede operation } i \text{ of job } j;\ j = 1, \dots, n; h, i = 1, \dots, m;$$

$$C_{j_i} \geqslant p_{0 j_i} \qquad \text{j=1, \dots ,n; i=1, \dots ,m;}$$

$$C_{k_i} \geqslant C_{j_i} + p_{j_i, k_i} \quad \text{or} \quad C_{j_i} \geqslant C_{k_i} + p_{k_i, j_i} \qquad \text{for all jobs } j \neq k \text{ and all machines } i = 1, \dots, m.$$

The first set of constraints link m otherwise independent single machine problems. Balas concentrated on such single machine polyhedra, adding sequence–dependent processing times (but no precedence constraints) and gave characterizations of the vertices and extreme directions of the polyhedra $P_{n,\pi}$ associated with a job permutation π. He showed that, for this very general class of single machine scheduling problems (but in the absence of precedence constraints or deadlines), the unit vectors are the sole extreme directions of the convex hull of feasible schedules. He also showed the following general lifting result:

Theorem 2.16. [Bal85] *Let $S \subset N$ be two job sets, and $P(S)$ and $P(N)$ be the corresponding single machine polyhedra with sequence–dependent processing times. An inequality $\sum_{j \in S} \alpha_j C_j \geqslant b$ is facet defining for $P(S)$ iff it is also facet defining for $P(N)$.*

Thus, to show that a valid inequality with small support S is facet defining for $P(N)$, it is sufficient to show that it is facet defining for the smaller–dimensional polyhedron $P(S)$. This result, in particular, allowed Balas to characterize all facets induced by inequalities with at most three nonzero coefficients. For example, using the theory of blocking polyhedra (cf., [Ful70, Ful71]), he proved that the inequalities

$$C_j \geqslant p_{0j}$$

are facet defining for the single machine scheduling polyhedron. The reader should be able to prove this directly by a slight modification of the proof presented above for Proposition 2.1. While these inequalities are in a sense more general than (2.2), they lack many of the nice properties of the latter; for example Balas shows that there are up to four different facet defining inequalities with the same support of size three, whereas inequalities (2.2) are uniquely defined by their support.

Turning back to his original multiple machine (job shop) problem, Balas also presented a sufficient condition for a facet defining inequality for a single machine polyhedron to also define a facet of the job shop polyhedron.

2.7 Parallel machines

We now turn to the use of natural dates for formulating parallel machine scheduling problems. Note that, when we represent a schedule by a vector $C \in \mathbb{R}^n$ of job completion times, we choose to ignore the *assignment* of jobs to the machines. In several of these problems, such as those with identical parallel machines, a precise assignment of jobs to machines is relatively unimportant and can easily be reconstructed from the completion time vector of a feasible schedule.

Very little is known about the polyhedral structure of parallel machine scheduling problems. The trivial lower bounds

$$C_j \geqslant p_j \qquad \text{all } j \in N \tag{2.35}$$

and release date constraints (2.23) are easily shown to be facet inducing for the corresponding parallel machine scheduling polyhedra without precedence constraints. Note, however, that for the dominant scheduling polyhedron for m identical parallel machines without precedence constraints, no facet inducing inequality $aC \geqslant b$ (with $a \geqslant 0$ for validity) can have support $S(a) = \{j \in N : a_j > 0\}$ of size $1 < |S(a)| \leqslant m$. (Indeed, if $1 < |S(a)| \leqslant m$, then the minimum of aC is obtained by scheduling the jobs in $S(A)$ each on a different machine; the resulting inequality is then implied by the trivial lower bound or release date constraints.) The extension of the parallel inequalities (2.2) to parallel machines is nontrivial; in fact, the problem of minimizing their left hand side $\sum p_j C_j$ is already NP-hard for two identical parallel machines [LRKB77].

Parallel machine scheduling polyhedra are simple and well understood in the case of *unit jobs*, i.e, when all $p_j = 1$, see [Sch93a]. Queyranne and Schulz [QS95] consider an extension to machines with different speeds, where these speeds may vary over time and across machines at different rates. Such machine–dependent speed functions may be used to model such predictable effects as operator learning and tool wear and tear, as well as such planned activities as machine upgrades, maintenance and the preassignment of other operations, all of which may affect the available processing speed of the different machines at different points in time. The authors consider jobs with identical processing requirements (a natural extension of unit jobs to this context) and allow so–called *compatible* release dates, which may occur only at instants where all machines are available to start a job in any undominated schedule. Special cases include integral release dates for identical parallel machines; and a common release date for uniform machines (each with constant speed). They show that the convex hull of feasible completion time vectors is a supermodular polyhedron. This implies that a weighted sum $\sum w_j C_j$ of completion times can be minimized by a greedy algorithm.

As for the case of release dates, we will now see that some simple relaxations in natural date variables, which have nicely structured optimal solutions, are sufficient to yield interesting approximation results. We now return to scheduling on m identical parallel machines, and establish a simple (though fairly weak) generalization of the parallel inequalities (2.2).

Lemma 2.17. [Sch96b, HSSW96] *The completion time vector C of any feasible schedule on m identical parallel machines satisfies*

$$\sum_{j \in S} p_j C_j \geqslant \frac{1}{2m} p(S)^2 + \frac{1}{2} p^2(S) \qquad \textit{for all } S \subseteq N. \tag{2.36}$$

Proof. Consider any feasible schedule and let C denote its completion time vector. For any job subset $S \subseteq N$ and any machine $i \in \{1, \ldots, m\}$, let S_i denote the set of all jobs from S processed on machine i. The parallel inequalities (2.2) apply to each machine i, implying

$$\sum_{j \in S} p_j C_j = \sum_{i=1}^{m} \sum_{j \in S_i} p_j C_j \geqslant \sum_{i=1}^{m} \frac{1}{2} \left(p(S_i)^2 + p^2(S_i) \right) = \frac{1}{2} \sum_{i=1}^{m} p(S_i)^2 + \frac{1}{2} p^2(S). \tag{2.37}$$

The result now follows from the fact that, with $\sum_{i=1}^{m} p(S_i) = p(S)$, the sum of squares $\sum_{i=1}^{m} p(S_i)^2$ is at its minimum $\frac{1}{m} p(S)^2$ when every $p(S_i)$ is equal to $\frac{1}{m} p(S)$. $\square$

The *proof* of the following theorem makes use of the weaker inequalities

$$\sum_{j\in S} p_j C_j > \frac{1}{m}\Big(\frac{1}{2}\big(p(S)^2 + p^2(S)\big)\Big) \qquad \text{for all } S \subseteq N, \tag{2.38}$$

which define a supermodular polyhedron, and of the nice structure of optimal solutions to a linear program defined with these constraints.

Theorem 2.18. [HSSW96] *Let C^{LP} denote an optimal solution to the LP relaxation*

$$\min \sum_{j\in N} w_j C_j \tag{2.39}$$

$$\text{subject to (2.35) and (2.36)} \tag{2.40}$$

of the scheduling problem $\mathsf{P}||\sum w_j C_j$. *Then*

$$wC^{LP} > \frac{1}{2}\big(1 + \frac{1}{2m-1}\big) wC^* \tag{2.41}$$

where C^* denotes any optimal schedule.

Proof. First consider the relaxation LPW of the LP (2.39)–(2.40) whereby all the constraints are replaced by the weaker inequalities (2.38). Note that the right hand sides of these inequalities (2.38) are precisely those of the single machine parallel inequalities (2.2) scaled by $\frac{1}{m}$. Thus the feasible set of LPW is also an affine image of a supermodular polyhedron, and LPW is solved by a greedy algorithm: rank the jobs in WSPT order $w_1/p_1 \geqslant w_2/p_2 \geqslant \ldots \geqslant w_n/p_n$ and let $C_j^{LPW} = \frac{1}{m} p(\{1,\ldots,j\})$. Let C^{WSPT} denote the completion time vector of the feasible schedule defined by Graham's list scheduling rule [Gra66] using this WSPT order; namely, every job is considered in the WSPT order and is then assigned to the earliest available machine. By the nature of Graham's list scheduling rule, we have, for all $j = 1,\ldots,n$, the following job-by-job bounds:

$$C_j^{WSPT} \leqslant \frac{1}{m} p(\{1,\ldots,j-1\}) + p_j = C^{LPW} + (1 - \frac{1}{m}) p_j. \tag{2.42}$$

Since C^{WSPT} is a feasible schedule, $w \geqslant 0$, LPW is a relaxation to LP, and by (2.35), we then have

$$wC^* \leqslant wC^{WSPT} \leqslant wC^{LPW} + (1 - \frac{1}{m}) \sum_{j\in N} w_j p_j \leqslant wC^{LP} + (1 - \frac{1}{m}) wC^{LP}$$

and the result follows. □

Note also that the WSPT list scheduling rule used in the proof of Theorem 2.18 was analyzed by Kawaguchi and Kyan [KK86], who proved that $wC^{WSPT} \leqslant \frac{1}{2}(\sqrt{2}+1) wC^*$. As observed by Hall et al., from inequality (2.42) and $C^* \geqslant p_j$, we obtain here a simple proof of the weaker bound $C^{WSPT} \leqslant (2 - \frac{1}{m}) wC^*$.

Before turning to identical parallel machine scheduling with release dates, we state witout proof an immediate generalization of Lemma 2.9:

Lemma 2.19. *Assume that vector $C \in \mathbb{R}^n$ satisfies $C_1 \leqslant C_2 \leqslant \ldots \leqslant C_n$ and all the inequalities* (2.36). *Then*

$$C_j > \frac{1}{2m} \sum_{i=1}^{j} p_i \qquad \textit{for all } j = 1,\ldots,n. \tag{2.43}$$

Consider now identical parallel machine scheduling with release dates. Note that Graham's list scheduling rule [Gra66] extends to this case by considering each job j in the list order and then inserting it in the earliest block of p_j consecutive idle time units after the release date r_j, without disturbing any already scheduled job.

Theorem 2.20. [Sch96b] *Let C^{LP} denote an optimal solution to the LP relaxation*

$$\min \sum_{j \in N} w_j C_j \tag{2.44}$$

$$\text{subject to (2.23) and (2.36)} \tag{2.45}$$

of the scheduling problem $P|r_j|\sum w_j C_j$ *(where all $w_j \geqslant 0$). Let C^H denote the completion time vector of the feasible schedule obtained by Graham's list scheduling rule using the order of the components in C^{LP}. Then*

$$wC^{LP} > \frac{1}{4}\Big(1+\frac{1}{4m-1}\Big)wC^* \quad \textit{and} \quad wC^H < \Big(4-\frac{1}{m}\Big)wC^* \tag{2.46}$$

where C^* denotes any optimal schedule.

Proof. As in earlier proofs, assume that $C_1^{LP} \leqslant C_2^{LP} \leqslant \cdots \leqslant C_n^{LP}$, and, for any given j, let $S = \{1, \ldots, j\}$. Note that, in schedule C^H, all machines must be busy between time $r_{\max}(S) = \max_{k \in S} r_k$ and the start $C_j^H - p_j$ of job j. Thus

$$C_j^H \leqslant r_{\max}(S) + \frac{1}{m} p(\{1, \ldots, j-1\}) + p_j = r_{\max}(S) + \frac{1}{m} p(S) + \Big(1-\frac{1}{m}\Big) p_j.$$

For $k \in \arg\max_{j \in S} r_j$, we have $r_{\max}(S) < C_k^{LP} \leqslant C_j^{LP}$. Lemma 2.19 implies $\frac{1}{m}p(S) \leqslant 2C_j^{LP}$. The release date constraints (2.23) and $r_j \geqslant 0$ imply $p_j \leqslant r_j + p_j \leqslant C_j^{LP}$. Therefore we obtain the job-by-job bound $C_j^H \leqslant (4-\frac{1}{m})C_j^*$ for all $j \in N$, and the results follow. □

We suspect that the bounds in Theorem 2.20 are not tight. We also note that Hall et al. [HSSW96] and Chakrabarti et al. [CPS+96] prove that somewhat more sophisticated approximation techniques show that natural date formulations also give constant approximation ratios for the case of identical parallel machines in the presence of precedence constraints (with and without release dates). Finally, Schulz [Sch96b] extends these ideas and proof techniques to m-stage flowshop problems, obtaining approximation bounds of $2m$ without release dates and $2m+1$ with release dates; in both cases, the bounds apply with or without precedence constraints. When there are parallel machines at each stage, the bounds change to $3m$ and $3m+1$, respectively, this time without precedence constraints.

Additional Notes and References

When *preemption* is allowed, it turns out that the convex hull of feasible completion time vectors is in general no longer closed. This was already observed in Section 2.1 for nonpreemptive scheduling with sequence–dependent processing times (see Figure 1). Queyranne and Wang [QW92] showed that this is also the case for preemptive single machine scheduling with either deadlines or precedence constraints. This property is intimately linked to whether or not there is an advantage to preemption when minimizing a weighted sum $\sum w_j C_j$ of completion times. When there is no advantage to preemption, valid inequalities for nonpreemptive schedules are also valid for preemptive schedules (the converse being always true, as preemptive problems are relaxations of their nonpreemptive counterparts). When this is the case, the convex hull of feasible preemptive schedules coincides with that of nonpreemptive schedules. Using this observation, Queyranne and Wang [QW91b] showed that in the precedence case this convex hull is closed if and only if the precedence constraints form an out–forest, since it then coincides with the convex hull of nonpreemptive schedules.

The release date and precedence inequalities (2.23) and (2.9), the parallel and shifted parallel inequalities (2.2) and (2.31), however, all remain valid for preemptive schedules. Therefore the LP relaxations described in preceding sections and which only use these inequalities, are also relaxations for the preemptive versions of the respective scheduling problems. Therefore all the resulting bounds also apply to the preemptive problems. Improving upon Corollary 2.14 for the preemptive case, Hall et al. [HSSW96] show that a preemptive version of the list scheduling heuristic using the order of completion times in an optimal solution to the LP relaxation in Corollary 2.14, yields a 2–approximation for the preemptive problem 1|r_j,prec,pmtn|$\sum w_jC_j$. They also show a similar result for the parallel machine problem P|r_j,prec,prmtn|$\sum w_jC_j$ and obtain a 3–approximation; if all release dates $r_j = 0$, their algorithm is a $(3-\frac{1}{m})$–approximation. Finally, note that for single or parallel machine problems (with integral release dates), the preemptive algorithms do not use any preemptions in the case of unit jobs, i.e., when all $p_j = 1$. Thus all the preceding preemptive results apply to the nonpreemptive unit job problems.

The notion of the generalized permutahedron of a poset, defined in Section 2.2 above, has recently been extended by Schrader, Schulz and Wambach [SSW96] to the case of a general, strictly supermodular function. Let $D = (N,A)$ be a poset and let $f : 2^N \to \mathbb{R}$ be a strictly supermodular set–function. We know from the greedy algorithm for supermodular polyhedra that there is a one–to–one correspondence between the vertices, say x^π, of the base polytope $B(f)$ and the permutations π of N. In view of the relation of the generalized permutahedron to base polytopes we may be interested in the convex hull of those vertices of $B(f)$ that correspond with a permutation that extends D. We denote this polytope by $B(f,D)$, i.e.,

$$B(f,D) := \text{conv}\{x^\pi : \pi \text{ is a linear extension of } D\} \ .$$

Schrader et al. [SSW96] show that, for a series–parallel poset D, this polytope $B(f,D)$ is completely described by the inequalities $x(I) \geqslant f(I)$ for all ideals I and a suitable extension of the series inequalities. Moreover, results similar to Theorem 2.2, Corollary 2.3, and Theorem 2.7 can be proved simply by using the strict supermodularity of f.

Some extensions of the results related to supermodular polyhedra were also obtained for stochastic and dynamic scheduling problems (cf., [FG86, FG88b, FG88a, SY92, BGT92, BNM93]). In many multiclass queueing systems, strong conservation laws are satisfied by certain performance measures. These laws imply that the feasible space of achievable performance is the base of a polymatroid. Therefore, many types of functions of the performance vector can be efficiently optimized. Bertsimas and Niño–Mora [BNM93] showed that this remains true if the performance measure satisfies a generalized conservation law by observing that in this case the performance space can be seen as an extended polymatroid (cf., [BGT92]). Notice that the polytope B_n is itself an extended polymatroid.

3 Time–Indexed Variables

The general task of machine scheduling is to allocate machines to jobs over time. Therefore, many scheduling problems are naturally formulated as integer programs with variables indexed by pairs (j,t) where j denotes a job and t is a time period. Such formulations are commonly referred to as *time–indexed* formulations [Sou89, DW90, SW92, AHS93, Akk94].

3.1 Problem Formulation and Complexity

In order to obtain a finite number of variables, we introduce a fixed time horizon (planning horizon) T and discretize time into, say, the periods $1,2,\dots,T$ where *period* t starts at time $t-1$ and ends at time t. (Remember that we assume all processing times p_j and all release dates and deadlines, if applicable, to be integral.) If we define the incidence vector x^F indexed by the pairs (j,t), $j \in N$, $t = 1,\dots,T$, of a feasible schedule F by

$$x^F_{jt} := \begin{cases} 1, & \text{if job } j \text{ is started in period } t; \\ 0, & \text{otherwise,} \end{cases}$$

the object of interest is the *time–indexed polytope*

$$TI := \operatorname{conv}\{x^F : F \text{ feasible schedule}\} \ .$$

This is a $0/1$ polytope in the $(n \cdot T)$–dimensional space. Even if we reduce the number of variables (e. g., by observing that x_{jt} has to be zero if $t > T - p_j + 1$, since job j cannot start after time $T - p_j$ without exceeding the planning horizon), the number of variables remains in general huge in comparison to the number of jobs, the input dimension. Notice that $T \geqslant p(N)$, at least when dealing with a single machine. That is, even if we would somehow derive a strongly polynomial time algorithm for optimizing a linear objective function over the polytope TI, this would only lead to a pseudo–polynomial algorithm for the original scheduling problem. We will later discuss one method used to deal algorithmically with this number of variables.

The precise problem setting that has been investigated in the literature and that we want to discuss here is as follows. We are given a single machine which can execute at most one job at a time. Preemption is not allowed and each job j has to be processed for a period of integral length p_j on this machine. We also require that every job starts at an integral time. Using time–indexed variables, this leads to the following formulation where c_{jt} is the cost for starting job j in period t:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt} \\
\text{subject to} \quad & \sum_{t=1}^{T-p_j+1} x_{jt} = 1 \qquad j = 1,\dots,n, && (3.1)\\
& \sum_{j=1}^{n} \sum_{s=t-p_j+1}^{t} x_{js} \leqslant 1 \qquad t = 1,\dots,T, && (3.2)\\
& x_{jt} \geqslant 0 \qquad j = 1,\dots,n,\ t = 1,\dots,T-p_j+1, && (3.3)
\end{aligned}$$

and we assume that $x_{jt} = 0$ for those $t \notin \{1,\dots,T-p_j+1\}$. Each integer solution to this linear system is the incidence vector of a feasible schedule, i. e., TI is identical to the integral hull of the polytope defined by the system (3.1) – (3.3). Equations (3.1) ensure that each job is scheduled exactly once whereas inequalities (3.2) take care that at most one job is executed in each period. In case $T = p(N)$, the inequalities in (3.2) can be replaced by the corresponding equations.

The chosen variables allow handling a broad collection of different models. Job release dates r_j or deadlines $\overline{d}_j$ are handled by setting $x_{jt} := 0$ for $t \leqslant r_j$ or $t > \overline{d}_j - p_j + 1$, respectively. If we want job j to precede job k in each feasible schedule, we may enforce this by setting $x_{k1} = \dots = x_{kp_j} = x_{j,(T-p_j-p_k+2)} = \dots = x_{j,(T-p_j+1)} = 0$ and by adding either the inequalities

$$\sum_{s=p_j+1}^{t} x_{ks} - \sum_{s=1}^{t-p_j} x_{js} \leqslant 0 \qquad t = p_j+1,\dots,T-p_k+1 \tag{3.4}$$

or the following inequality proposed by Sousa [Sou89]:

$$\sum_{t=p_j+1}^{T-p_k+1} (t-1)x_{kt} \geqslant \sum_{t=1}^{T-p_j-p_k+1} (t-1)x_{jt} + p_j \ . \tag{3.5}$$

We show later (see Lemma 3.9) that a feasible solution to (3.1) – (3.4) also satisfies (3.5).

Different objective functions can be formulated using an appropriate choice of the costs c_{jt}. In particular, all standard min–sum criteria are linear in the time–indexed variables. Choosing $c_{jt} := w_j(t-1)$ we minimize the weighted sum of starting times, whereas $c_{jt} := w_j \max(0, t+p_j-d_j-1)$ leads to the minimization of the total weighted tardiness. Here d_j stands for the due date of job j. The weighted number of late jobs is minimized by using $c_{jt} := w_j$ if $t > d_j - p_j + 1$ and $c_{jt} := 0$, otherwise. Since minimizing the total weighted tardiness is already strongly NP–hard for a single machine (see [Law77] or [LRKB77]), the considered scheduling problem is strongly NP–hard in general. Thus, we cannot hope for finding a complete description of TI in terms of linear equations and inequalities (cf., e. g., [Sch86] for a thorough discussion of this subject). However, if all the jobs have unit processing time, i. e., $p_j = 1$ for $j \in N$, then the constraint matrix of (3.1) and (3.2) is the node–edge incidence matrix of a bipartite graph, and is therefore totally unimodular. As a consequence, the described scheduling problem, and thus all the special cases mentioned above (with the exception of those involving precedence constraints), are solvable in polynomial time in this case. This remains true when all jobs have the same processing time p and $T = n \cdot p + c$ for some constant c. In this case the scheduling problem can be solved by solving $\binom{n+c}{c}$ assignment problems (cf., [CS95] for details). On the other hand, if all jobs have the same processing time $p_j = 2$ and T is part of the input, Crama and Spieksma [CS95] showed that the scheduling problem is NP–hard, even if all $c_{jt} \in \{0, 1\}$.

3.2 Dimension and Basic Facets

Notice that we already know that the time–indexed polytope TI is not of full dimension. Since equations (3.1) are linearly independent, we can conclude that $\dim(TI) \leqslant \sum_j (T - p_j + 1) - n = nT - \sum_j p_j$. Under the assumption $T \geqslant \sum_j p_j + p_{\max}$ where $p_{\max} := \max\{p_j : j \in N\}$, Sousa and Wolsey [SW92] showed that equality holds.

Theorem 3.1. [SW92] *If* $T \geqslant \sum_{j \in N} p_j + p_{\max}$, *then* $\dim(TI) = nT - \sum_{j \in N} p_j$.

Proof. We include a proof because it is simpler than the original one. We show that the equations (3.1) define a minimal equation system for TI. Our proof generalizes the one of Crama and Spieksma [CS95] which is restricted to jobs with identical processing times. We already mentioned that system (3.1) is valid for TI and that its matrix has full row rank. Thus, it remains to be shown that for any equation

$$\sum_{k=1}^{n} \sum_{s=1}^{T-p_k+1} \alpha_{ks} x_{ks} = \beta \tag{3.6}$$

satisfied by all feasible schedules there exists $\lambda \in \mathbb{R}^N$ such that $\lambda_k = \alpha_{ks}$ for all $s = 1, \ldots, T - p_k + 1$. Let j be a job and $t \in \{1, \ldots, T - p_j\}$ a time period. Because of the assumption on T there exists a feasible schedule x such that $x_{jt} = 1$ and $x_{ks} = 0$ for all jobs $k \in N \setminus \{j\}$ and all time periods $s \in \{t, t+1, \ldots, t+p_j\}$. Let $\bar{x}$ be the schedule with all jobs in the same position as in x but with job j starting in period $t+1$. Observe that $\bar{x}$ is also feasible. Therefore, we obtain

$$0 = \beta - \beta = \sum_{k=1}^{n} \sum_{s=1}^{T-p_k+1} \alpha_{ks} x_{ks} - \sum_{k=1}^{n} \sum_{s=1}^{T-p_k+1} \alpha_{ks} \bar{x}_{ks} = \alpha_{jt} - \alpha_{j,(t+1)} \ .$$

With $\lambda_k := \alpha_{k1}$ equation (3.6) is a linear combination of the equations in (3.1). □

A slight modification of this proof (see, e. g., [Sch96a]) shows that all nonnegativity constraints (3.3) induce facets of TI.

Theorem 3.2. *If $T \geqslant \sum_{j \in N} p_j + p_{\max}$, then the inequalities $x_{jt} \geqslant 0$ define facets of TI, for all $j = 1, \ldots, n$ and all $t = 1, \ldots, T - p_j + 1$.*

In a similar way one can show that all the inequalities (3.2) define facets of TI (see Crama and Spieksma [CS95] for the restriction to identical processing times).

3.3 More Valid Inequalities

After providing an initial valid (integer) formulation of the considered problem, the next task is to find inequalities valid for the time–indexed polytope TI which strengthen the initial linear relaxation and therefore can contribute to a better lower bound for the original scheduling problem. Fix a job j and a time period t, and define $p^j_{\max} := \max_{k \neq j}\{p_k\}$. If we choose $\Delta \in \{2, \ldots, p^j_{\max}\}$, it follows from the restriction that jobs are not allowed to overlap, that no job $k \neq j$ can be started in one of the periods $t - p_k + \Delta, \ldots, t$ when the processing of job j starts in one of the periods $t - p_j + 1, \ldots, t + \Delta - 1$. On the other hand, no two jobs k and ℓ different from j can be started simultaneously in one of the periods $t - p_k + \Delta, \ldots, t$ and $t - p_\ell + \Delta, \ldots, t$, respectively. Thus, we have established the following observation.

Lemma 3.3. [SW92] *Let j be a job, t a time period, and $\Delta \in \{2, \ldots, p^j_{\max}\}$. Then the inequality*

$$\sum_{s=t-p_j+1}^{t+\Delta-1} x_{js} + \sum_{\substack{k=1 \\ k \neq j}}^{n} \sum_{s=t-p_k+\Delta}^{t} x_{ks} \leqslant 1 \tag{3.7}$$

is valid for the time–indexed polytope TI.

Observe that these inequalities are not implied by the system (3.1) – (3.3), except for $\Delta = 1$ in which case we get back inequalities (3.2). Sousa and Wolsey [SW92] derived this class of inequalities by applying a more general technique. Their idea is to sum up all the inequalities in the initial formulation, here inequalities (3.2), where the inequalities get a certain nonnegative weight. The polytope defined as the integral hull of the nonnegativity constraints, the aggregated constraint and the equation system is not only a relaxation of the original one (TI) but has also a special structure. It is defined by one knapsack constraint and generalized upper bound (GUB) constraints, i. e. the binary variables are partitioned into subsets (associated with each job) for which at most one variable can have value 1. Any inequality valid for this polytope is also valid for the original one, in particular GUB–cover inequalities (cf. [SW92, Wol90b]). In essentially this way Sousa and Wolsey obtained two more classes of valid inequalities for TI (see [SW92] for a detailed description of these inequalities). But in contrast to the family of inequalities (3.7) those are in general not very strong [AHS93]. If the time horizon T is large enough, Sousa and Wolsey established the facet defining property for inequalities (3.7). Van den Akker, van Hoesel and Savelsbergh [AHS93] showed that these inequalities together with the inequalities (3.2) provide (in a certain sense) all facet defining inequalities of TI that have integral coefficients and right–hand side 1. (Crama and Spieksma [CS95] derived the same result for the special case of identical jobs.) In fact, van den Akker, van Hoesel and Savelsbergh consider the convex hull of characteristic vectors of *partial schedules* where the constraint that each job has to be executed exactly once is replaced by the constraint that each job has to be executed at most once. We call this polytope the *extended submissive* of TI and denote it by TI^-_e. An initial valid formulation whose integral hull is TI^-_e is given by

$$\sum_{t=1}^{T-p_j+1} x_{jt} \leqslant 1 \quad \text{for } j = 1, \ldots, n, \tag{3.8}$$

together with (3.2) and (3.3).

The extended submissive of the time–indexed polytope TI has the technical advantage to be full–dimensional. In particular, the origin and all unit vectors belong to TI_e^-. However, TI_e^- cannot replace TI for the purpose of optimization but can be helpful for the investigation of the facial structure of TI. To give an example, observe that it is quite easy to find $\sum_j (T - p_j + 1)$ linearly independent points in TI_e^- satisfying inequality (3.7) with equality. That is, (3.7) defines a facet of the extended submissive of TI. Notice that $TI = TI_e^- \cap \{x : \sum_{t=1}^{T-p_j+1} x_{jt} = 1 \text{ for } j = 1,\dots,n\}$ implies that a complete list of inequalities defining facets of TI_e^- contains for each facet of TI itself an inducing inequality.

In order to describe their result mentioned above more detailed we adopt some notation from van den Akker, van Hoesel and Savelsbergh [AHS93]. We denote by V the support of a valid inequality for $\operatorname{esub}(TI)$, and by V_j the set of those time periods t with $(j,t) \in V$. Furthermore, let $l_j := \min\{t : t - p_j + 1 \in V_j\}$ and $u_j := \max\{t : t \in V_j\}$, and let l (u) be the minimum (maximum) over all l_j (u_j). Finally, the interval $[s,t]$ stands for the set of time periods $\{s+1,\dots,t\}$.

A $0/1$ polytope P is called *down–monotone* if $P = \operatorname{conv}(S)$ for some set $S \subseteq \{0,1\}^n$ such that $x \leqslant y \in S$ and $x \in \{0,1\}^n$ imply $x \in S$. Since TI_e^- is a full–dimensional down–monotone $0/1$ polytope, it follows from a result of Hammer, Johnson and Peled [HJP75] that the nonnegativity constraints are the only facet defining inequalities with right–hand side 0, and that any other facet inducing inequality $ax \leqslant \beta$ with integral coefficients a_{jt} satisfies $\beta > 0$ and $a_{jt} \in \{0,1,\dots,\beta\}$. In particular, we know that any inequality that is facet defining for TI_e^-, that has integral coefficients, and that has right–hand side 1 is of the form $x(V) \leqslant 1$. Van den Akker, van Hoesel and Savelsbergh gave a more precise characterization.

Theorem 3.4. [AHS93] *An inequality $x(V) \leqslant 1$ which defines a facet of TI_e^- has the following structure*:

$$\begin{aligned} V_1 &= [l - p_1, u], \\ V_j &= [u - p_j, l], \qquad j \in \{2,\dots,n\}, \end{aligned}$$

where $l = l_1 \leqslant u_1 = u$.

Theorem 3.4 states in particular that all the sets V_j are (possibly empty) intervals. Since each facet inducing inequality $x(V) \leqslant 1$ is maximal in the sense that there does not exist any $W \supset V$ such that $x(W) \leqslant 1$ is valid, it is quite easy to observe that all facet inducing inequalities for TI_e^- with right–hand side 1 are given by (3.7), (3.8), and (3.2). Thus, we can conclude that all inequalities that define facets of TI, that have integral coefficients and right–hand side 1, and that are also valid for TI_e^- are contained in the classes (3.2) and (3.7). Van den Akker, van Hoesel and Savelsbergh [AHS93] derive also characterizations of inequalities that define facets of TI_e^- and have right–hand side 2.

3.4 Computational Experiences

Sousa and Wolsey [SW92] as well as Crama and Spieksma [CS95] report on first computational experiences with a branch&cut algorithm based on the inequalities they derived. Both authors studied randomly generated instances with different objective function types. Since they do not present any elegant separation procedure, we omit the details of their algorithms. But we want to mention a technique which is useful in any branch&cut algorithm but almost essential when dealing with a polytope in a such high–dimensional space like TI. Using a good upper bound U on the optimal objective function value which can be derived from a primal heuristic, some nonbasic variables of the current LP solution that are either on their lower bound 0 or on their upper bound 1 can be fixed forever at this bound. Let L be the objective function value of the current LP solution. If x_{jt} is nonbasic and has reduced cost $\overline{c}_{jt}$, we can fix x_{jt} to zero if $x_{jt} = 0$ and $\overline{c}_{jt} > U - L$, and we can fix x_{jt} to one if $x_{jt} = 1$ and $\overline{c}_{jt} < L - U$. Using this technique in order to reduce the number of variables, Sousa and Wolsey as well as Crama and Spieksma are able to solve problem instances with up to 30 jobs and with $p_{max} \leqslant 5$.

Because of the major drawback of the time–indexed approach, namely the large number of variables as well as the large number of initial constraints that lead immediately to large memory requirements and computation times, van den Akker [Akk94] applied Dantzig–Wolfe decomposition [DW60] in order to solve the initial LP–relaxation by column generation. If we denote the polytope defined by the inequalities (3.2) and the nonnegativity constraints (3.3) by PS the initial LP–relaxation can be viewed as:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt} \\ \text{subject to} \quad & \sum_{t=1}^{T-p_j+1} x_{jt} = 1 \qquad j = 1, \ldots, n \\ & x \in PS. \end{aligned}$$

Van den Akker observed that the polytope PS is integral since the constraint matrix belonging to the inequalities (3.2) is an interval matrix and therefore totally unimodular. The vertices of PS are "schedules", van den Akker called them *pseudo–schedules* where each job can be assigned to less than, to more than, or to exactly one starting time. But the machine does never execute more than one job at a time. If $x^1, x^2, \ldots, x^K$ are the extreme points of PS, each $x \in PS$ can be written as $x = \sum_{r=1}^{K} \lambda_r x^r$ where $\sum_{r=1}^{K} \lambda_r = 1$ and $\lambda_r \geqslant 0$. Replacing x in the above formulation by this expression we obtain the initial LP–relaxation in its *master form*:

$$\begin{aligned} \text{minimize} \quad & \sum_{r=1}^{K} \left(\sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt}^{r} \right) \lambda_r \\ \text{subject to} \quad & \sum_{r=1}^{K} \left(\sum_{t=1}^{T-p_j+1} x_{jt}^{r} \right) \lambda_r = 1 \qquad j = 1, \ldots, n, \\ & \sum_{r=1}^{K} \lambda_r = 1 \\ & \lambda_r \geqslant 0 \qquad\qquad r = 1, \ldots, K. \end{aligned}$$

This master problem has only $n+1$ constraints but involves in general a huge number of variables. If we are able to solve the pricing problem quite efficiently, this does not matter since a basis contains only $n+1$ variables. The pricing problem is to determine a column with negative reduced cost, if one exists, or to find out that this is not the case. That is, the pricing problem is the separation problem for the dual LP. Van den Akker observed that in our case the pricing problem can be solved in time $O(nT)$ by determining a shortest path from the source to the target in the acyclic network associated with the above mentioned interval matrix supplemented by T unit columns which encode the associated tree of this network matrix (cf. Schrijver [Sch86] for a detailed discussion of network matrices). Notice that there is an one–to–one correspondence between the source–target–paths in this network and the pseudo–schedules. Van den Akker reports on the solution of instances of the LP–relaxation with up to 30 jobs and $p_{max} \leqslant 100$ using column generation. She also shows that in this case it is easy to combine column with row generation (see [BJN$^+$94] for a general treatment of this topic). Since facet inducing inequalities with right–hand side 1 or 2 are formulated in the original variables, they only influence the objective in the pricing problem but not its structure. Her algorithm based on row and column generation works for problems ranging in size from 20 to 30 jobs with $p_j \leqslant 50$ and $p_j \leqslant 30$, respectively.

If the overall solution of the column generation algorithm is fractional one has to proceed with branch&bound to end up with an optimal solution for the scheduling problem. The idea is to handle the obtained subproblems in the same way as the root problem. Thus, altogether this leads to a *branch&price* algorithm. See Barnhart et al. [BJN+94] for a general survey of branch&price, and Desrosiers et al. [DDSS94] for applications of branch&price in time constrained routing and scheduling.

3.5 Relations to the Generalized Permutahedron

The time–indexed variables studied in this section allow, in particular, an integer linear programming formulation of the nonpreemptive single machine problem with the objective to minimize the weighted sum of completion times $\sum_j w_j C_j$. Since we studied this problem in Section 2 above in terms of natural date variables, it is natural to consider relationships between the associated polyhedra. It is obvious that, if $T = p(N)$, the generalized permutahedron B_n is the affine image of the time–indexed polytope TI under the mapping

$$C_j = \sum_{t=1}^{T-p_j+1} (t-1)\, x_{jt} + p_j \ , \qquad j = 1, \ldots, n. \tag{3.9}$$

It is less obvious that the generalized permutahedron is also the affine image of the relaxation of TI that is defined by (3.1) – (3.3) under the same mapping. The observation needed to verify this claim is stated in the following lemma.

Lemma 3.5. [Sou89] *For* $1||\sum w_j C_j$ *and nonnegative weights* w_j*, the value of an optimal solution to the linear relaxation* (3.1) – (3.3) *with objective function* $\sum_{j=1}^{n} w_j \big(\sum_{t=1}^{T-p_j+1} (t-1) x_{jt} + p_j \big)$ *is equal to the value of an optimal schedule.*

This result was also proved by Van den Akker [Akk94] and by Chan et al. [CMSL96].

Lemma 3.5 implies that each point C defined by (3.9) where x is a solution to (3.1) – (3.3) satisfies the parallel inequalities (2.2). Consequently, we can draw the following conclusion.

Theorem 3.6. *For* $T = p(N)$*, the generalized permutahedron* B_n *is precisely the image of the polytope defined by* (3.1) – (3.3) *under the affine mapping given by* (3.9)*. For* $T = \infty$*, the dominant* P_n *of the generalized permutahedron is precisely the image of the polytope defined by* (3.1) – (3.3) *under the affine mapping given by* (3.9).

Proof. Assume first that $T = p(N)$. Theorem 2.4 states that B_n is completely described by the parallel inequalities (2.2) and equation (2.3). It follows from Lemma 3.5 that the image of a solution to (3.1) – (3.3) under the mapping (3.9) satisfies the parallel inequalities. To show that equation (2.3) is also satisfied we argue as follows.

First, we rewrite the expression for C_j using equation (3.1) and $x_{j,(T-p_j+2)} = \cdots = x_{jT} = 0$:

$$\begin{aligned}
C_j &= \sum_{t=1}^{p(N)} (t-1)\, x_{jt} + p_j \\
&= \sum_{t=1}^{p(N)} (t-1)\, x_{jt} + p_j + \sum_{t=1}^{p(N)} x_{jt}(1+2+\cdots+(p_j-1)) - \frac{1}{2}p_j(p_j-1) \\
&= \frac{1}{p_j}\Big(p_j \sum_{t=1}^{p(N)} (t-1)\, x_{jt} + \sum_{t=1}^{p(N)} x_{jt}(1+2+\cdots+(p_j-1)) - \frac{1}{2}p_j(p_j-1)\Big) + p_j \\
&= \frac{1}{p_j}\Big(\sum_{t=1}^{p(N)} x_{jt}((t-1)+t+\cdots+(t+p_j-2)) - \frac{1}{2}p_j(p_j-1)\Big) + p_j \\
&= \frac{1}{p_j}\Big(\sum_{t=1}^{p(N)} (t-1) \sum_{s=t-p_j+1}^{t} x_{js} - \frac{1}{2}p_j(p_j-1)\Big) + p_j \ .
\end{aligned}$$

Then, we use this new expression to obtain

$$\begin{aligned}
\sum_{j\in N} p_j C_j &= \sum_{j\in N}\Big(\sum_{t=1}^{p(N)} (t-1) \sum_{s=t-p_j+1}^{t} x_{js} - \frac{1}{2}p_j(p_j-1)\Big) + p^2(N) \\
&= \sum_{t=1}^{p(N)} (t-1) \sum_{j\in N} \sum_{s=t-p_j+1}^{t} x_{js} + \frac{1}{2}p^2(N) + \frac{1}{2}p(N) \\
&= \frac{1}{2}p(N)(p(N)-1) + \frac{1}{2}p^2(N) + \frac{1}{2}p(N) \\
&= \frac{1}{2}p(N)^2 + \frac{1}{2}p^2(N) \ ,
\end{aligned}$$

where we used (3.2) in equation form for the last but one equality.

The claim for $T = \infty$ follows directly from Lemma 3.5 since P_n is completely described by the parallel inequalities (2.2). □

Corollary 3.6 generalizes the well–known fact that the permutahedron is an affine image of the assignment polytope.

From Corollary 3.6 we can conclude that the value obtained by optimizing the weighted sum of completion times over the relaxation of TI defined by (3.1) – (3.3) where $T = p(N)$ is equal to the optimal value, even if we do not impose any restriction on the weights. The following example shows, however, that this relaxation of the time–indexed polytope has in general optimal vertices that are fractional.

Example 3.7. Consider an instance with three jobs such that $p_1 = 2$, $p_2 = 2$, $p_3 = 3$, and $w_1 = 2$, $w_2 = 2$, and $w_3 = 3$. Since we are concerned with minimizing the total weighted completion time each sequence is optimal and has value 33. Observe that $x_{11} = x_{13} = x_{24} = x_{26} = x_{31} = x_{35} = \frac{1}{2}$ defines a feasible solution to (3.1) – (3.3) which has objective function value 33, too. Routine calculations show that x is a vertex of the polytope defined by (3.1) – (3.3). In fact, if we change the objective function by adding a big–M to the coefficients associated with the job–time pairs $(1,4)$, $(1,6)$, $(2,1)$, $(2,3)$, and $(3,3)$ we already see that x has better objective function value than all feasible schedules.

Van den Akker [Akk94] shows that for unit weights, i. e., $w_j = 1$ for all jobs j, the set of optimal solutions to the LP relaxation (3.1) – (3.3) is a face of the time–indexed polytope TI, that is, *all* optimal extreme points of the time-indexed formulation are integral.

We mentioned above that release dates can be taken care of in the time–indexed formulation by fixing certain variables to zero. This has the following implication.

Lemma 3.8. *Let x be a solution to the linear relaxation* (3.1) – (3.3) *of the single machine scheduling problem and assume that $x_{j1} = \cdots = x_{jr_j} = 0$, for some job j and its release date r_j. Then the point C defined by* (3.9) *satisfies all parallel inequalities and, in addition, $C_j \geqslant r_j + p_j$.*

Proof. The fixing of variables implies

$$C_j = \sum_{t=1}^{T-p_j+1} (t-1)x_{jt} + p_j = \sum_{t=r_j+1}^{T-p_j+1} (t-1)x_{jt} + p_j \geqslant r_j \sum_{t=r_j+1}^{T-p_j+1} x_{jt} + p_j = r_j + p_j \ ,$$

where we used (3.1) for the last equality. □

Let us rest for a moment on the strongly NP–hard problem to minimize the total weighted completion time on a single machine subject to release dates, i. e., $1|r_j|\sum w_j C_j$. We have two different linear programming relaxations at the hand. There is the relaxation in completion time variables given by the parallel inequalities (2.2) and $C_j \geqslant r_j + p_j$ for all $j \in N$ on one side, and the relaxation in time–indexed variables given by (3.1) – (3.3) and the fixing of variables on the other side. Whereas the formulation in completion time variables relies on a separation routine the time–indexed formulation suffers from the large number of variables. Theorem 3.6 and Lemma 3.8 imply that the lower bound obtained from the time–indexed formulation is at least as good as the one obtained from the formulation in completion time variables. However, as we already observed in Subsection 2.5 we may strengthen the parallel inequalities by "shifting" the entire schedule by the smallest release date of some subset of jobs. That is, the following inequalities are valid:

$$\sum_{j\in S} p_j C_j \geqslant r_{\min}(S) \sum_{j\in S} p_j + \frac{1}{2}\Big(\sum_{j\in S} p_j^2 + \big(\sum_{j\in S} p_j\big)^2\Big) \qquad \text{for all } S \subseteq N. \tag{3.10}$$

The separation problem associated with the class of inequalities (3.10) can be solved in polynomial time, see [QS95] for details. Moreover, recently Goemans [Goe96] pointed out that the polyhedron defined by the inequalities (3.10) is a supermodular polyhedron. Hence, the lower bound can be computed directly. Goemans also showed that the lower bound obtained in this way coincides exactly with the lower bound obtained from solving the following relaxation:

$$\begin{aligned}
\text{minimize} \qquad & \sum_{j=1}^{n} w_j C_j \\
\text{subject to} \qquad & \sum_{j=1}^{n} y_{jt} \leqslant 1 && t = 0,1,\dots,T, \\
& \sum_{t=0}^{T} y_{jt} = p_j && j = 1,\dots,n, \\
\frac{p_j}{2} + \frac{1}{p_j}\sum_{t=0}^{T}\Big(t+\frac{1}{2}\Big)y_{jt} &= C_j && j = 1,\dots,n, \\
& y_{jt} \geqslant 0 && j = 1,\dots,n,\ t = r_j, r_j+1,\dots,T.
\end{aligned} \tag{3.11}$$

The time–indexed variables y have the meaning $y_{jt} = 1$ if job j is being processed in the time period $[t, t+1]$ and $y_{jt} = 0$, otherwise. Quite interestingly, Dyer and Wolsey [DW90] observed that the lower bound obtained from solving the latter relaxation (or equivalently, the one defined by the shifted parallel inequalities) is at most as strong as the one obtained from solving (3.1) – (3.3) when release dates are taken into account by fixing variables. Consequently, the latter relaxation is the best known for $1|r_j|\sum w_jC_j$ in terms of the quality of bounds but it is not of polynomial size. Note that Proposition 2.12 implies that all these relaxations are 3-relaxations.

Finally, we discuss the case in which jobs are not independent. If there are precedence constraints, each feasible solution x to the LP relaxation (3.1) – (3.3) which satisfies in addition the inequalities (3.4) also satisfies the simple precedence inequalities in terms of completion time variables.

Lemma 3.9. *Assume that job j has to precede job k in every feasible schedule. Let x be a point contained in the polytope defined by the linear system* (3.1) – (3.4). *Moreover, assume that $x_{j,(T-p_j-p_k+2)} = \cdots = x_{jT} = 0$ and $x_{k1} = \cdots = x_{kp_j} = 0$. Then, the point $C \in \mathbb{R}^N$ defined by* (3.9) *satisfies $C_k \geqslant C_j + p_k$.*

Proof. We may write the difference $C_k - C_j$ as follows:

$$
\begin{aligned}
C_k - C_j &= \sum_{t=p_j+1}^{T-p_k+1} (t-1)x_{kt} - \sum_{t=1}^{T-p_j-p_k+1} (t-1)x_{jt} + p_k - p_j \\
&= \sum_{t=p_j+1}^{T-p_k+1} \big((t-1)x_{kt} - (t-p_j-1)x_{j,(t-p_j)}\big) + p_k - p_j \\
&= \sum_{t=p_j+1}^{T-p_k+1} p_j x_{kt} + \sum_{t=p_j+1}^{T-p_k+1} (t-p_j-1)(x_{kt} - x_{j,(t-p_j)}) + p_k - p_j \\
&= p_k + \sum_{t=p_j+1}^{T-p_k+1} (t-p_j-1)(x_{kt} - x_{j,(t-p_j)}) \ ,
\end{aligned}
$$

where we used equation (3.1) for the last equality. To show that the last expression $\sum_{t=p_j+1}^{T-p_k+1}(t-p_j-1)(x_{kt} - x_{j,(t-p_j)})$ is nonnegative, we proceed as follows. We start backwards. From $x_{k,(T-p_k+1)} = \sum_{t=p_j+1}^{T-p_k+1} x_{kt} - \sum_{t=p_j+1}^{T-p_k} x_{kt}$ and $x_{j,(T-p_j-p_k+1)} = \sum_{s=1}^{T-p_j-p_k+1} x_{js} - \sum_{s=1}^{T-p_j-p_k} x_{js}$ together with equation (3.1) and inequality (3.4) follows that $x_{k,(T-p_k+1)} \geqslant x_{j,(T-p_j-p_k+1)}$. Hence, we may keep the last summand $(T-p_k-p_j)(x_{k,(T-p_k+1)} - x_{j,(T-p_j-p_k+1)})$ in the considered expression as a credit for forthcoming terms. Now, we are interested in the last but one summand. Again, from (3.1) and (3.4) follows $x_{k,(T-p_k)} + x_{k,(T-p_k+1)} - x_{j,(T-p_j-p_k+1)} \geqslant x_{j,(T-p_j-p_k)}$. Consequently, the sum of the last two summands of the considered expression is nonnegative. By continuing in the same manner, $\sum_{t=p_j+1}^{T-p_k+1}(t-p_j-1)(x_{kt} - x_{j,(t-p_j)}) \geqslant 0$ follows. □

Corollary 2.14 shows that the time-indexed relaxation of $1|r_j,\text{prec}|\sum w_jC_j$ is again a 3-relaxation.

3.6 Identical Parallel Machines

The approach of using time–indexed variables also enables us to formulate scheduling problems that involve more than just one machine as an integer linear programming problem. For example, if we are given m identical parallel machines (instead of a single machine) on each of which every job can be processed, the following straightforward formulation is valid:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt} \\
\text{subject to} \quad & \sum_{t=1}^{T-p_j+1} x_{jt} = 1 \qquad j = 1, \ldots, n, && (3.12) \\
& \sum_{j=1}^{n} \sum_{s=t-p_j+1}^{t} x_{js} \leqslant m \qquad t = 1, \ldots, T, && (3.13) \\
& x_{jt} \in \{0, 1\} \qquad j = 1, \ldots, n, \; t = 1, \ldots, T - p_j + 1, && (3.14)
\end{aligned}$$

where x_{jt} is defined exactly as before. Note that, in this formulation, we do not specify the assignment of jobs to the parallel machines.

This formulation has a nice scaling property, which we now define in a fairly general context. Consider an n-job, m-machine instance of a scheduling problem, and a relaxation (R) : $z^R = \min\{dy : y \in Y\}$, where y is a vector of variables, Y is a corresponding feasible set, and d defines a linear objective function. By "relaxation", we mean that z^R is a lower bound on the optimal value of the given scheduling problem. Let $y \in Y$ be a rational feasible solution, and M an integer such that My is integral. Now, consider the Mn-job, Mm-machine *multiplied instance* obtained by making M identical copies of each original job and M identical parallel copies of each original machine. We say that the relaxation (R) has the *scaling property* if My defines a feasible schedule with value Mdy for the multiplied instance.

To verify that the time-indexed formulation (3.12) – (3.14) has this scaling property, consider any rational solution x to its LP relaxation, and let M be the least common multiple of all the denominators in x. Then solution Mx defines a schedule for the multiplied instance, where to every fractional component $x_{jt} = \frac{\mu}{M}$ of x we associate μ copies of job j, which start in period t on μ parallel machines. It is then easily verified that this schedule is feasible for the multiplied instance and has value Mcx.

The scaling property is used by van den Akker [Akk94] and by Schulz [Sch96a] to show that the LP relaxation to the time-indexed formulation (3.12) – (3.14) solves the unit–weight problems $1||\sum C_j$ and $P||\sum C_j$, respectively, and by Chan et al. [CMSL96] to analyze the optimality gap for a different relaxation to $P||\sum w_j C_j$. Our presentation below uses the general framework introduced by Chan et al.

A *Parameter List Scheduling Heuristic* (PLSH) is a scheduling method for a parallel machine problem, whereby an *index* is computed for each job on the basis of its parameters (e.g., processing time p_j, weight w_j) only; the jobs are then sequenced in nondecreasing (or nonincreasing) order of this index, by assigning the next job to the machine that becomes first available. Classic examples of PLSH's are the Shortest Processing Time, or SPT rule (whereby the jobs are sequenced in nondecreasing order of their processing times) and the Weighted Shortest Processing Time, or WSPT rule (whereby the jobs are sequenced in nondecreasing order of their weight to processing times ratios w_j/p_j). In a multiplied instance of $P||\sum w_j C_j$, as defined above, we may assume that a PLSH will sequence all M copies of a same job consecutively, and will assign them to the M copies of a same machine. Letting z^H and z_M^H denote the objective value of the resulting solutions in the original and multiplied instances, respectively, we have $z_M^H = M z^H$. Finally we say that a heuristic H is an α*–approximation* if, for all instances, the value z^H of the solution it produces is no worse than α times the optimal value z^*.

Lemma 3.10. *If (R) is a relaxation to a scheduling problem and has the scaling property, and if there exists an α–approximation Parameter List Scheduling Heuristic for this scheduling problem, then $z^R \geqslant \frac{1}{\alpha} z^*$, where $\alpha \geqslant 1$ and z^* denotes the optimal value of the scheduling problem.*

Proof. Let x^R denote an optimal solution to relaxation (R) with value $z^R = dx^R$. Using the scaling property of (R), let $z_M^R = Mz^R$ denote the value of the corresponding feasible schedule in the multiplied instance, where M is an appropriate scaling parameter. We have

$$Mz^* \leqslant Mz^H = z_M^H \leqslant \alpha z_M^* \leqslant \alpha z_M^R = \alpha M z^R$$

where the first and last inequalities follow from comparing an optimal value with that of a feasible solution, and the second inequality from the α–approximation property applied to the multiplied instance. □

Using the fact that the SPT rule is optimal (i.e., $\alpha = 1$) for the unit–weight problem $P||\sum C_j$ (see [LLRKS93] for references), we obtain:

Corollary 3.11. [Sch96a] *The optimal value of the parallel machine, total completion time problem* $P||\sum C_j$ *coincides with the optimal value of the LP relaxation to the time-indexed formulation* (3.12) – (3.14).

As mentioned earlier, the special case of Corollary 3.11 for a single machine is due to Van den Akker [Akk94]. Turning now to general weights, we may use the WSPT heuristic (whereby jobs are sorted in nonincreasing order of their w_j/p_j ratios), which Kawaguchi and Kyan [KK86] show to be a $\frac{\sqrt{2}+1}{2}$–approximation heuristic for parallel machines:

Corollary 3.12. [CMSL96] *The optimal objective value of the LP relaxation to the time-indexed formulation* (3.12) – (3.14) *for* $P||\sum w_j C_j$ *is within a factor of at least* $\frac{2}{\sqrt{2}+1} \approx 0.828$ *of the optimum.*

Chan et al. show corresponding results for a set partitioning formulation, and also consider some nonlinear convex objective functions. In addition, they show that the optimal value of the LP relaxation to time-indexed formulation (3.12) – (3.14) for $P||\sum w_j C_j$ equals that of another, slightly weaker, set partitioning formulation, and that the latter is asymptotically optimal when the job weights and processing times (w_j, p_j) form a random sample from a (joint) probability distribution (W,P), under the mild technical assumption that the expected value of the product WP is finite. Note that this latter result applies to a common method of generating instances for computational experiments, and explains the very small optimality gaps observed in [CP95] and [AHV].

3.7 Additional Notes and References

In an *open shop*, each job must visit all the m machines, but the order of these visits is not specified in advance. We may think of job j as consisting of m operations $j_1, j_2, \dots, j_m$ with processing times $p_{j_1}, p_{j_2}, \dots, p_{j_m}$ associated with machines $1, 2, \dots, m$, respectively. Thus, we have the same problem as discussed above with operations instead of jobs and the additional constraint that operations belonging to the same job must not overlap. Each solution to the following system is the incidence vector of a feasible schedule:

$$\sum_{t=1}^{T} x_{j_i,t} = 1 \qquad j = 1,\dots,n, \; i = 1,\dots,m, \tag{3.15}$$

$$\sum_{j=1}^{n} \sum_{s=t-p_{j_i}+1}^{t} x_{j_i,s} \leqslant 1 \qquad t = 1,\dots,T, \; i = 1,\dots,m, \tag{3.16}$$

$$\sum_{i=1}^{m} \sum_{s=t-p_{j_i}+1}^{t} x_{j_i,s} \leqslant 1 \qquad t = 1,\dots,T, \; j = 1,\dots,n, \tag{3.17}$$

$$x_{j_i,t} \in \{0,1\} \qquad j = 1,\dots,n, \; i = 1,\dots,m, \; t = 1,\dots,T. \tag{3.18}$$

This formulation naturally emphasizes the symmetric structure of the open shop problem: we may interchange jobs and machines. However, if the order in which the operations of a job have to pass through the machines is prescribed, we lose that symmetry but are still able to model this *job shop* problem by replacing the inequalities (3.17) by the following precedence constraints

$$\sum_t (t-1)\, x_{j_{\sigma_j(i+1)},t} \geqslant \sum_t (t-1)\, x_{j_{\sigma_j(i)},t} + p_{j_{\sigma(i)}} \qquad j = 1,\dots,n, \;\; i = 1,\dots,m-1,$$

when the linear ordering σ_j represents the specified order of the operations of job j.
So far, the polyhedral structure of the convex hull of the integer solutions of such multiple machine formulations has not been investigated.

We finally come back to the single machine problem. If one deals with a regular performance measure, and if in addition all jobs have the same release time, it is known for a long time [CMM67] that there exists an optimal schedule without induced machine idle time. This implies that the overall completion time is equal to $p(N)$. We pointed out that in this case the inequalities (3.2) of the initial formulation can be turned into equations. And we used this in the proof of Theorem 3.6. However, most of the other results for the polytope TI itself were obtained under the assumption that the time horizon T exceeds $p(N)$ sufficiently. This assumption is acceptable for optimization, but only theoretically. When implementing a branch&cut procedure based on this time–indexed formulation one wishes to use the smallest possible value for T, namely $T = p(N)$ and to use inequalities which are strong in this case. Therefore, a thorough investigation of TI in this case might be worth doing.

In their work on approximation algorithms for single machine scheduling problems, Hall, Shmoys, and Wein [HSW96] (see also [HSSW96]) had a nice idea to overcome the difficulty with the exponential number of time–indexed variables. They subdivide the time horizon at geometrically increasing points so that one can focus on jobs completing within intervals, rather than at specific times. Since the completion and the start time of each interval have a bounded ratio, one can assign a job to complete within an interval without too much concern about when within the interval it actually completes. For an arbitrarily small positive constant ε let $\tau_0 = 1$ and $\tau_\ell = (1+\varepsilon)^{\ell-1}$. The single machine relaxation is then:

$$\begin{array}{lrcll}
\min & \sum_{j=1}^n w_j \sum_{\ell=1}^L \tau_{\ell-1} x_{j\ell} & & & \\
\text{s. t.} & \sum_{\ell=1}^L x_{j\ell} & = & 1 & \text{for all } j \\
 & x_{j\ell} & = & 0 & \text{if } \tau_\ell < r_j + p_j \\
 & \sum_{s=1}^\ell x_{js} - \sum_{s=1}^\ell x_{ks} & \geqslant & 0 & \text{if } (j,k) \in A, \text{ for all } \ell \\
 & \sum_{s=1}^\ell \sum_{j=1}^n p_j x_{jk} & \leqslant & \tau_\ell & \text{for all } \ell \\
 & x_{j\ell} & \geqslant & 0 &
\end{array}$$

Notice that here $x_{j\ell} = 1$ means that job j *completes* in the ℓ-th interval. One unusual aspect of this formulation is that it is the linear relaxation of an integer program that is, itself, a relaxation of the original problem.

After the discussion of the polyhedral approach to scheduling using x_{jt}–variables one may think of other variables also indexed by pairs (j,t) but with a different meaning. Almost completely analogous to the x_{jt}–variables are variables that have value 1 if job j is completed at time t and value 0, otherwise. Lawler [Law64] used those variables to obtain an assignment formulation for the scheduling problem on m identical and parallel machines, and jobs with identical processing times. If we change this problem by allowing for arbitrary integer processing times and preemption, we know that it is enough to permit splitting of a job j in maximal p_j time units. Again Lawler [Law64] observed that this problem can be modeled using the following time–indexed variables:

$$y_{jt} = \begin{cases} 1, & \text{if one unit of job } j \text{ is assigned to period } t; \\ 0, & \text{otherwise.} \end{cases}$$

The formulation is

$$\sum_t y_{jt} = p_j \qquad j \in N$$
$$\sum_j y_{jt} \leqslant m \qquad \text{for all } t$$
$$y_{jt} \in \{0,1\}$$

Since this is an assignment problem, each of the extreme points of its linear programming relaxation is integral.

4 Linear Ordering Variables

In many machine scheduling problems the set of feasible solutions can be considered as (a subset of) the set of permutations, or *linear orderings* of the job set N. Hence one possibility for describing the feasible solutions is to use variables that characterize linear orderings. In this section, we investigate and review formulations that use $0/1$ variables δ^{π}_{jk} for pairs of jobs $j,k \in N$, and permutations $\pi : N \to \{1,\dots,n\}$ with the meaning

$$\delta^{\pi}_{jk} = \begin{cases} 1, & \text{if } j \text{ precedes } k \text{ in } \pi; \\ 0, & \text{otherwise.} \end{cases}$$

4.1 The Linear Ordering Polytope

The convex hull of all linear orderings described by these δ–incidence vectors is known as the *linear ordering polytope* P_{LO}. This polytope has been studied since 1953, and seems to belong to the best understood class of polyhedra associated with hard combinatorial optimization problems, see Fishburn [Fis92] for an overview. Because the *linear ordering problem* (given weights c_{jk} for all $j,k \in N, j \neq k$, find a linear ordering π that minimizes $\sum_{j,k,j\neq k} c_{jk}\delta^{\pi}_{jk}$) is NP–hard, it may be difficult to obtain a complete description of P_{LO} in terms of linear equations and inequalities. However, many classes of facet inducing inequalities are known (see, e. g., [GJR85, Fis92]). An initial formulation, the integer solutions of which are exactly the incidence vectors of linear orderings, is given by

$$\delta_{jk} + \delta_{kj} = 1 \quad \text{for } 1 \leqslant j < k \leqslant n, \tag{4.1}$$
$$\delta_{jk} + \delta_{k\ell} - \delta_{j\ell} \leqslant 1 \quad \text{for } j,k,\ell \in N, j \neq k \neq \ell \neq j, \tag{4.2}$$
$$\delta_{jk} \geqslant 0 \quad \text{for } j,k \in N, j \neq k, \tag{4.3}$$

where constraints (4.2) are known as *transitivity* constraints (or as *triangle* constraints when stated in the equivalent form $\delta_{jk} + \delta_{k\ell} + \delta_{\ell j} \leqslant 2$).

The nonpreemptive single machine scheduling problem to minimize the weighted sum of completion times, $1||\sum w_j C_j$, can be modeled as a linear program on P_{LO}. This is due to the fact that given a permutation via its incidence vector δ, the completion time of job j in the corresponding tight permutation schedule (without idle time) is given by

$$C_j = \sum_{\substack{k\in N\\ k\neq j}} p_k \delta_{kj} + p_j \; . \tag{4.4}$$

The objective function then becomes

$$\sum_{j\in N} w_j C_j = \sum_{\substack{j,k\in N\\ k\neq j}} w_j p_k \delta_{kj} + \sum_{j\in N} w_j p_j \tag{4.5}$$

as was already observed by Potts [Pot80]. Notice that the cost coefficients have a very special structure. For the sequencing problem itself, without any other restrictions, Peters [Pet88] as well as Nemhauser and Savelsbergh [NS92] showed, using Smith's rule [Smi56] and linear programming duality, that we may drop the integrality as well as the transitivity constraints (4.2). Then the optimal objective value of the linear program with objective function (4.5) and constraints (4.1) and (4.3) equals the value of a minimum weight schedule. We now present an even simpler derivation of this result, which implies Smith's rule. First, we drop half of the variables δ_{jk} by use of (4.1), keeping only those with $j < k$, and also drop the transitivity constraints (4.2), to obtain the relaxation

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{1\leqslant j<k\leqslant n}(w_k p_j - w_j p_k)\delta_{jk} + \sum_{1\leqslant j<k\leqslant n} w_j p_k + \sum_{j\in N} w_j p_j & \\ \text{subject to} & 0 \leqslant \delta_{jk} \leqslant 1 \qquad \text{for } 1 \leqslant j < k \leqslant n. & \end{array}$$

Since minimizing $\sum_{1\leqslant j<k\leqslant n}(w_k p_j - w_j p_k)\delta_{jk}$ over the hypercube in $\mathbb{R}^{n(n-1)/2}$ is easy, we obtain

$$\delta_{jk} = \begin{cases} 1, & \text{if } w_j/p_j \geqslant w_k/p_k \\ 0, & \text{otherwise} \end{cases}$$

as an optimal solution (breaking ties consistently with the transitivity constraints (4.2)). Notice that this solution is the characteristic vector of a linear ordering. This observation implies Smith's rule. This result also implies that each vector C defined by (4.4) where δ is a solution to (4.1) and (4.3) (without the transitivity constraints) satisfies the parallel inequalities (2.2). Moreover, it also satisfies equation (2.3). We combine these observations in the following theorem.

Theorem 4.1. *Let B_n denote the convex hull of feasible completion time vectors* (*see Section* 2.2). *Then, the generalized permutahedron B_n is an affine image of*

(a) *the hypercube in $\mathbb{R}^{n(n-1)/2}$;*

(b) *the polytope defined by the constraints* (4.1) *and* (4.3)*;*

(c) *the polytope defined by the system* (4.1) – (4.3)*;*

(d) *the linear ordering polytope P_{LO}.*

Proof. The affine mapping $L : \mathbb{R}^{n(n-1)} \to \mathbb{R}^n$ to consider for (b), (c), and (d) is given by (4.4) (for (a) a straightforward adaptation works). Whereas the inclusion $B_n \subseteq L(P_{\text{LO}})$ is obvious, the other inclusions follow from our observations above and Theorem 2.4. □

4.2 The Linear Extension Polytope

In the presence of precedence constraints several permutations turn out to be infeasible. The *linear extension polytope* $P_{\text{LO}}(D)$ is the convex hull of all δ^{π}–vectors where π are the linear extensions of the given poset $D = (N, A)$. Equivalently, $P_{\text{LO}}(D)$ is the integral hull of the polytope defined by the following linear system.

$$\begin{array}{rcll} \delta_{jk} + \delta_{kj} & = & 1 & \text{for } 1 \leqslant j < k \leqslant n \text{ with } j \parallel k, \qquad (4.6) \\ \delta_{jk} + \delta_{k\ell} - \delta_{j\ell} & \leqslant & 1 & \text{for } j,k,\ell \in N \text{ with } j \neq k \neq \ell \neq j, \qquad (4.7) \\ \delta_{jk} & \geqslant & 0 & \text{for } j,k \in N \text{ with } j \parallel k, \qquad (4.8) \\ \delta_{jk} & = & 1 & \text{for } j,k \in N \text{ with } (j,k) \in A, \qquad (4.9) \\ \delta_{jk} & = & 0 & \text{for } j,k \in N \text{ with } (k,j) \in A. \qquad (4.10) \end{array}$$

Here $j \parallel k$ means that the ordering relation between j and k is not determined in advance. The following lemma and its proof extend the analogous ones for the linear ordering polytope due to Grötschel, Jünger, and Reinelt [GJR85].

Lemma 4.2. *Let $D = (N,A)$ be a poset. Then the equations* (4.6), (4.9)*, and* (4.10) *form a minimal linear equation system defining the affine hull of $P_{\text{LO}}(D)$.*

Proof. Let $d\delta = d_0$ be an arbitrary equation that is satisfied by all linear extensions of D. Observe that we only have to show that $d_{jk} = d_{kj}$ for all $j \parallel k$. If $j \parallel k$, there exists a linear extension π of D with $\pi(k) = \pi(j) + 1$. Then π' which arises from π by letting $\pi'(j) := \pi(k)$, $\pi'(k) := \pi(j)$, and $\pi'(\ell) := \pi(\ell)$, otherwise, is also a linear extension of D. Thus, we conclude that $0 = d\delta^{\pi} - d\delta^{\pi'} = d_{jk} - d_{kj}$. Finally, the minimality of this system follows from the fact that every variable appears in exactly one equation. □

Maybe surprisingly, the fixing (4.9) and (4.10) of the variables enforcing the precedence constraints ensures a bound on the objective function value of an instance of $1|\text{prec}|\sum w_j C_j$ that is at least as good as the one obtained by using the natural date approach, including all parallel and all series inequalities. This is based on the following observation.

Lemma 4.3. *Each vector C defined by* (4.4)*, where δ is a solution to* (4.6) – (4.10) *satisfies the series inequalities* (2.20).

For a proof, the reader is referred to [Sch96a].

An interesting consequence of Lemma 4.3 is that we can optimize in polynomial time over a polytope, namely the affine image of the one defined by (4.6) – (4.10), that approximates the generalized permutahedron $B_n(D)$ at least as well as the polytope defined by equation (2.3), by all parallel inequalities (2.2) and by all series inequalities (2.20). Recall that no separation algorithm is known for the class of series inequalities. Lemma 4.3, however, implies that the series inequalities are contained in a class of inequalities that (by linear programming duality) can be separated in polynomial time. We summarize the implications of Theorem 4.1 and Lemma 4.3 in the following theorem.

Theorem 4.4.

(a) *If the precedence relation D is series–parallel, $B_n(D)$ is precisely the image of the polytope defined by* (4.6) – (4.10) *under the affine mapping* (4.4).

(b) *If D is series–parallel, then*

$$\min\Big\{ \sum_{\substack{j,k\in N\\ j\neq k}} w_j p_k \delta_{kj} : \delta \textit{ satisfies } (4.6) – (4.10)\Big\} \quad = \quad \min\Big\{ \sum_{\substack{j,k\in N\\ j\neq k}} w_j p_k \delta_{kj} : \delta \in P_{\text{LO}}(D)\Big\} \ .$$

(c) *There exists a polynomial time algorithm that, for any poset $D = (N,A)$, and for any point $C \in \mathbb{Q}^N$, either asserts that C satisfies all parallel and series inequalities, or finds an inequality violated by C from a class of valid inequalities that contains the parallel and series inequalities that define facets of $B_n(D)$.*

(d) *For any poset D, the image of* (4.6) – (4.10) *under the affine mapping* (4.4) *is contained in the polytope defined by the parallel and the series inequalities.*

Proof. (a) Theorem 4.1 and Lemma 4.3 state that the image of any feasible solution to (4.6) – (4.10) under the affine mapping (4.4) satisfies the parallel and series inequalities and equation (2.3). Theorem 2.11 implies that this image therefore is contained in $B_n(D)$. The other inclusion is trivial. Statement (b) follows immediately from (a). Since system (4.6) – (4.10) is of polynomial size, it follows from Theorem 4.1 and Lemma 4.3 that we can optimize over a polytope in polynomial time that approximates $B_n(D)$ as least as well as the one defined by all parallel and all series inequalities. Since optimization and separation are equivalent with respect to polynomial time solvability [GLS88], it follows that we can solve the separation problem associated with a polytope that is contained in the one defined by all parallel and series inequalities in polynomial time. Observe that the parallel and series inequalities that define facets of $B_n(D)$ also define facets of this polytope. In other words, the facet defining parallel and series inequalities are contained in a class of inequalities that can be separated in polynomial time. This shows (c). The last claim (d) again follows immediately from Theorem 4.1 and Lemma 4.3. □

Thus, in case of series–parallel precedence constraints we have two different LP formulations of the sequencing problem with weighted sum of completion time objective, namely the natural date one and the one in linear ordering variables. Each vertex of the generalized permutahedron $B_n(D)$ is indeed the completion time vector of a feasible schedule whereas the linear extension polytope $P_{\mathrm{LO}}(D)$ may have optimal non–integer vertices. For example, without precedence constraints and in the case of unit weights and unit processing times, i.e., $w_j = p_j = 1$ for all j, *every* feasible solution to (4.1) – (4.3) is optimal with respect to the objective function (4.5), as follows directly from adding all equations (4.1). However, the system (4.1) – (4.3) is not sufficient to describe the linear ordering polytope P_{LO}.

Wolsey [Wol90a] studied formulations of the single machine problem with precedence constraints in the (C,δ)–space. He derived a class of $\mathrm{O}(n^2)$ inequalities that can replace the transitivity constraints in the series–parallel case. If $(j,k) \in A$, a straightforward way to connect the completion time variables C_j and C_k with the linear ordering variables δ_{jk} is the nonlinear inequality

$$C_k - C_j \geqslant \sum_{\ell \in N \setminus \{j,k\}} p_\ell \delta_{j\ell} \delta_{\ell k} + p_k \quad . \tag{4.11}$$

Wolsey proposed the following linear relaxation of inequality (4.11):

$$C_k - C_j \geqslant p_k + \sum_{\ell \in S(j) \cap P(k)} p_\ell + \sum_{\substack{\ell \in S(j) \setminus P(k) \\ \ell \neq k}} p_\ell \delta_{\ell k} + \sum_{\substack{\ell \in P(k) \setminus S(j) \\ \ell \neq j}} p_\ell \delta_{j\ell} \ , \tag{4.12}$$

where $P(k) := \{\ell \in N : (\ell,k) \in A\}$ and $S(j) := \{\ell \in N : (j,\ell) \in A\}$ denote the set of all predecessors of job k and the set of all successors of job j with respect to $D = (N,A)$, respectively. He also showed that the series inequalities (2.20) are dominated by a nonnegative combination of constraints of type (4.1) and (4.12). This has two (related) consequences. First, the LP lower bound we obtain by using the constraints (4.6), (4.8), (4.9), (4.10), and (4.12) as an LP relaxation for the scheduling problem with arbitrary precedences, is at least as good as the one obtained by using the natural date relaxation with all parallel inequalities (2.2) and all series inequalities (2.20). On the other hand, Theorem 2.11 implies the following result.

Theorem 4.5. [Wol90a] *If the precedence constraints D are series–parallel, the projection of* $Q := \{(C,\delta) \in \mathbb{R}^n \times \mathbb{R}^{n(n-1)} : (C,\delta)$ *satisfies* (4.4), (4.6), (4.8), (4.9), (4.10), *and* (4.12) $\}$ *into the C–space is exactly the generalized permutahedron $B_n(D)$.*

Note that there are $\mathrm{O}(n^2)$ constraints (4.12), as compared with $\mathrm{O}(n^3)$ transitivity constraints (4.7). However, whereas formulation Q is the most compact known LP formulation of the sequencing problem

$1|\text{prec}|\sum w_jC_j$ when the precedence order is series–parallel, it is in general not valid in the following sense. A point $(C,\delta) \in Q$ with $\delta \in \{0,1\}^{n(n-1)}$ defines not necessarily a schedule as shown by the following example.

Example 4.6. Consider an instance of $1|\text{prec}|\sum w_jC_j$ with five jobs that all have the same processing time. Assume that job 1 has to precede job 3 and that job 2 is a predecessor of jobs 3 and 4. The point defined by $\delta_{12} = \delta_{14} = \delta_{35} = 0$ and $\delta_{13} = \delta_{15} = \delta_{23} = \delta_{24} = \delta_{25} = \delta_{34} = \delta_{45} = 1$ satisfies (4.6), (4.8), (4.9), (4.10), and (4.12) but violates the transitivity constraint $\delta_{41} + \delta_{13} - \delta_{43} \leqslant 1$.

Since inequalities (4.12) are implied by constraints (4.6) – (4.10), the latter formulation is superior for general precedence constraints. The lower bounds obtained by using this formulation may be further improved by implicitly adding known inequalities that define high–dimensional faces of the linear ordering polytope, e. g., *Möbius Ladder* inequalities [GJR85] (see also [Sch96a]). On the other hand, the natural date approach needs only n variables in comparison to $O(n^2)$ linear ordering variables and we may add the spider inequalities. Since the number of variables plays often a crucial role in branch&cut algorithms, the natural date polyhedra may be preferable.

4.3 Release Dates

The single machine problem $1|r_j|\sum w_jC_j$ is strongly NP–hard [LRKB77], where we have replaced the precedence constraints by release times r_j at which job j becomes available for processing. Since the introduction of release dates may force machine idle time, the weighted sum of completion times cannot be modeled by linear ordering variables only. This is also different from the case of deadlines. Peters [Pet88] studied the deadline problem from a polyhedral point of view, using both linear ordering and starting time variables.

Dyer and Wolsey [DW90], Wolsey [Wol90a], and Nemhauser and Savelsbergh [NS92] deal with the problem with release dates. They propose valid inequalities linking natural date and linear ordering variables, but none of these authors report solving problems with more than 30 jobs using these inequalities in a branch&cut algorithm.

The idea of Dyer and Wolsey is to consider a hierarchy $(R_s)_{0\leqslant s\leqslant n}$ of relaxations of the original problem. Here, relaxation (R_s) is obtained by combining enumeration of initial sequences of size s with Smith's rule. More precisely, for fixed s $(0 \leqslant s \leqslant n)$ (R_s) is the problem to determine a sequence consisting of s jobs out of the job set N such that the weighted sum of completion times is minimized if the release dates of the chosen jobs are taken into account, and if the remaining $n-s$ jobs are sequenced after in WSPT order (ignoring their release dates). Each of these problems can be formulated as a linear program with n start time variables S_j, $n(n-1)$ linear ordering variables δ_{jk}, and $s!$ *initial sequence variables* y_σ, where $y_\sigma = 1$ if the s–sequence σ is sequenced first and $y_\sigma = 0$, otherwise. Obviously, the bound obtained by using (R_{s+1}) is superior to the one obtained from (R_s). Notice that (R_0) is precisely the problem we obtain by disregarding all release dates (and therefore solved by Smith's rule) whereas (R_n) coincides with the original problem. For $s = 1$, i. e. when only the release date of the first job in the sequence is considered, the linear program describing (R_1) is given by

$$
\begin{array}{rcll}
\sum_{j\in N} r_j y_j + \sum_{\ell\in N\setminus\{k\}} p_\ell \delta_{\ell k} & \leqslant & S_k & \text{for } k\in N \\
\sum_{j\in N} y_j & = & 1 & \\
\delta_{jk} - y_j & \geqslant & 0 & \text{for all } j,k\in N, j\neq k \\
\delta_{jk} + \delta_{kj} & = & 1 & \text{for } 1\leqslant j<k\leqslant n \\
\delta \in \mathbb{R}_+^{n(n-1)} & & y\in\mathbb{R}_+^n &
\end{array} \tag{4.13}
$$

Notice, that we use start time variables S_j instead of completion time variables. This does not make a real difference since $C_j = S_j + p_j$, but may be more convenient for the reader when consulting the original literature. Let Q_1 denote the polyhedron in variables $(S,\delta,y)\in\mathbb{R}^n\times\mathbb{R}^{n(n-1)}\times\mathbb{R}^n$ that is defined by system (4.13). Since we do not really need the y–variables (we just used them to derive easily a LP–formulation of (R_1)), it is a good idea to compute the projection of Q_1 into the (S,δ)–space. Dyer and Wolsey showed that this projection is described by the following system.

$$
\begin{array}{rcll}
r_j - \sum_{\ell\in N}(r_j - r_\ell)^+ \delta_{\ell,\alpha(\ell)} + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k} & \leqslant & S_k & \text{for } j,k\in N, \text{and } \alpha\in\mathcal{M}_N \\
\sum_{j\in N}\delta_{j,\alpha(j)} & \geqslant & 1 & \text{for } \alpha\in\mathcal{M}_N \\
\delta_{jk}+\delta_{kj} & = & 1 & \text{for } 1\leqslant j<k\leqslant n \\
\delta_{jk} & \geqslant & 0 & \text{for } j,k\in N, j\neq k
\end{array} \tag{4.14}
$$

Here, $\mathcal{M}_A$ for $A\subseteq N$ stands for the set of all mappings $\alpha: A\to A$ with $\alpha(j)\neq j$ for all $j\in A$, and $a^+ \doteq \max\{0,a\}$. We are not going to repeat the proof of this fact that is based as usually on determining the extreme rays of the projection cone. But we give a straightforward explanation of the validity of inequalities (4.14):

$$
\begin{array}{rcl}
S_k & \geqslant & \sum_{\ell\in N} r_\ell y_\ell + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k} \\
& = & r_j\sum_{\ell\in N} y_\ell - \sum_{\ell\in N}(r_j-r_\ell)y_\ell + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k} \\
& = & r_j - \sum_{\ell\in N}(r_j-r_\ell)y_\ell + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k} \\
& \geqslant & r_j - \sum_{\ell\in N}(r_j-r_\ell)^+ y_\ell + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k} \\
& \geqslant & r_j - \sum_{\ell\in N}(r_j-r_\ell)^+ \delta_{\ell,\alpha(\ell)} + \sum_{\ell\in N\setminus\{k\}} p_\ell\delta_{\ell k},
\end{array}
$$

where we used $\sum_{\ell\in N} y_\ell = 1$, $(r_j - r_\ell)\leqslant(r_j-r_\ell)^+$, and $y_\ell\leqslant\delta_{\ell,\alpha(\ell)}$ to obtain the last three equalities and inequalities, respectively. Dyer and Wolsey observed that inequalities (4.14) remain valid for every feasible schedule of our original problem encoded in starting time and linear ordering variables, even if we consider subsets of the job set N.

Theorem 4.7. [DW90] *The constraints*

$$\begin{aligned}
r_j - \sum_{l\in A}(r_j - r_l)^+\delta_{l,\alpha(l)} + \sum_{l\in A\setminus\{k\}} p_l\delta_{lk} &\leqslant S_k && \textit{for } j,k \in A \subseteq N, \textit{and } \alpha \in \mathcal{M}_A && (4.15)\\
S_j &\geqslant r_j && \textit{for } j \in N && (4.16)\\
\delta_{jk} + \delta_{kj} &= 1 && \textit{for } 1 \leqslant j < k \leqslant n && (4.17)\\
\delta_{jk} + \delta_{kl} - \delta_{jl} &\leqslant 1 && \textit{for } j,k,l \in N, j \neq k \neq l \neq j && (4.18)\\
\delta_{jk} &\in \{0,1\} && \textit{for } j,k \in N, j \neq k && (4.19)
\end{aligned}$$

give a valid formulation for the single machine scheduling problem with release dates.

The major disadvantage of Dyer and Wolsey's mixed integer programming formulation is the huge number of inequalities needed, in spite to n^2 variables. Nemhauser and Savelsbergh [NS92] observed that already the relaxation

$$S_k \geqslant (r_j + p_j)\delta_{jk} + \sum_{l<j, l\neq k} p_l(\delta_{jl} - \delta_{kl}) + \sum_{l>j, l\neq k} p_l\delta_{lk} \qquad \text{for } j,k \in N, j \neq k \qquad (4.20)$$

of the nonlinear inequality

$$S_k \geqslant (r_j + p_j)\delta_{jk} + \sum_{l\neq k,j} p_l\delta_{jl}\delta_{lk} \qquad \text{for } j,k \in N, j \neq k$$

that we really wanted to add may replace (4.15). Here, we assume the jobs to be in nondecreasing order of their release dates, i. e., $r_1 \leqslant r_2 \leqslant \cdots \leqslant r_n$. Thus, they obtained a mixed integer programming formulation for the single machine scheduling problem with release times that is compact, i. e. both, the number of variables and the number of constraints are polynomially bounded in the input dimension n.

By observing that the inequalities (4.20) consists of three parts — a base release time, a part dealing with all jobs that have a release time smaller than the base release time, and a part including all jobs that become available for processing after the base release time — Nemhauser and Savelsbergh constructed several other valid inequalities, mostly by establishing different base release times and strengthening the various parts. For example, observe that $p_l(\delta_{jl} - \delta_{kl})$ is dominated by $(r_j - r_l)(\delta_{jl} - \delta_{kl}) + (r_l + p_l - r_j)\delta_{lk}$, if $r_l < r_j$ but $r_l + p_l > r_j$. Also, the base release time part $(r_j + p_j)\delta_{jk}$ in (4.20) can be strengthened by replacing it by $r_j + p_j\delta_{jk}$ in case $j < k$, and by $r_k + (r_j + p_j - r_k)\delta_{jk}$ in case $j > k$. Thus, the inequalities (4.20) should be replaced by

$$\begin{aligned}
S_k \geqslant\ & r_j + p_j\delta_{jk} + \sum_{\substack{l < j, l \neq k,\\ r_l + p_l \leqslant r_j}} p_l(\delta_{jl} - \delta_{kl})\\
&+ \sum_{\substack{l < j, l \neq k,\\ r_l + p_l > r_j}} \big((r_j - r_l)(\delta_{jl} - \delta_{kl}) + (r_l + p_l - r_j)\delta_{lk}\big) + \sum_{l>j, l\neq k} p_l\delta_{lk} \quad 1 \leqslant j \leqslant k \leqslant n
\end{aligned}$$

$$\begin{aligned}
S_k \geqslant\ & r_k + (r_j + p_j - r_k)\delta_{jk} + \sum_{\substack{l < j, l \neq k,\\ r_l + p_l \leqslant r_j}} p_l(\delta_{jl} - \delta_{kl})\\
&+ \sum_{\substack{l < j, l \neq k,\\ r_l + p_l > r_j}} \big((r_j - r_l)(\delta_{jl} - \delta_{kl}) + (r_l + p_l - r_j)\delta_{lk}\big) + \sum_{l>j, l\neq k} p_l\delta_{lk} \quad 1 \leqslant k < j \leqslant n
\end{aligned}$$

to obtain a stronger formulation in the following sense. The lower bound on the sequencing problem with release dates that is obtained by use of the LP relaxation (4.16) – (4.18) together with these inequalities above is at least as good as the one obtained with (4.20).

Additional Notes and References

Dyer and Wolsey [DW90] compared the quality of different formulations of the single machine problem $1|r_j|\sum w_jC_j$ with release dates in terms of the lower bounds obtained from their LP relaxations. They prove that the time–indexed formulation (3.1) – (3.3) provides better bounds than some other relaxations. Recall that we deal with release dates in the time–indexed model by fixing some variables to zero. In particular, the lower bound obtained from the time–indexed formulation is always at least as good as that from the formulation including completion time and linear ordering variables, defined as follows:

$$\begin{array}{llcll} \text{minimize} & \sum_{j\in N} w_jC_j & & & \\ \text{subject to} & \sum_{j\in N\setminus\{k\}} p_j\delta_{jk} + p_k & \leqslant & C_k & k\in N, \\ & \delta_{jk}+\delta_{kj} & = & 1 & 1\leqslant j<k\leqslant n, \\ & C_j & \geqslant & r_j+p_j & j\in N, \\ & \delta_{jk} & \geqslant & 0 & j,k\in N, j\neq k. \end{array}$$

This should not be too surprising since the latter formulation is not even valid! Indeed, the transitivity constraints are missing. Hence, the result of Dyer and Wolsey should not be considered as the final word with respect to the bounding quality of the different formulations.

Applegate and Cook [AC91] compared two cutting plane algorithms for the job shop problem based on natural date variables only, and on natural date and linear ordering variables for each machine, respectively. They used several straightforward extensions of the inequalities we presented in this section and in Section 2. They report on quite large computational efforts for these algorithms, in particular in comparison with standard procedures for obtaining lower bounds. Especially, the approach that makes use of linear ordering variables leads to very large running times.

From the order–theoretic point of view (see Möhring and Radermacher [MR89]), the single machine problem is a special case of the resource–constrained scheduling problem where all two–element subsets of the job set N are forbidden sets, i. e., cannot be scheduled in parallel. Thus, the only interval orders that induce feasible schedules are linear extensions of the precedence order D. A straightforward extension of the linear ordering variable approach to resource–constrained scheduling is therefore the use of the interval order polytope. The *interval order polytope* is defined as the convex hull of all interval orders and is studied in detail in [Sch96a].

The feasible solutions of the resource constrained scheduling problem correspond with those integral points δ in the interval order polytope that satisfy in addition

$$\begin{array}{rcll} \delta_{jk} & = & 1 & \text{for } j,k\in N \text{ with } (j,k)\in A \\ \sum_{j,k\in F}\delta_{jk} & \geqslant & 1 & \text{for each forbidden set } F \end{array}$$

However, in order to model some of the standard performance measures, natural date variables are needed. The completion time variables can be linked to the interval order variables by using the constraints

$$C_j + p_k\delta_{jk} - (1-\delta_{jk}-\delta_{kj})p_j - x_{kj}M \leqslant C_k, \qquad j,k\in N, j\neq k.$$

The obvious disadvantage of these linking inequalities is the need for the big–M.

5 Positional Date and Assignment Variables

In certain scheduling problems, such as single machine and some flow shop problems, it is often natural to work with date variables that refer not to the original jobs, but to their position in the schedule. Let τ_κ (resp., γ_κ) denote the start time (resp., the completion time) of the κ–th job in the schedule. (For clarity, we are using Greek letters to refer to any positional variable or index.) This definition, of course, only makes sense when there is no ambiguity as to which job is the first ($\kappa = 1$), second ($\kappa = 2$), etc., last ($\kappa = n$) in the schedule. Thus, for example, the makespan on one machine is simply γ_n. The term *positional* was coined by Emmons [Emm87] to refer to position–dependent weights, and used by Hoogeveen and van de Velde [HV95a] for positional completion time variables. This term seems preferable to the more ambiguous terms *generic* [LQ92] and *generalized* [Hal86] used by earlier authors. We use the term *natural* to refer to problem elements that are not positional.

5.1 A Formulation Using Positional and Assignment Variables

Positional variables arise in a control theoretic view of scheduling problems, are naturally associated with assignment variables, and are useful to model certain positional elements (e. g., positional deadline or release date constraints; symmetric objective functions) in scheduling problems.

Lasserre and Queyranne [LQ92] view a nonpreemptive scheduling system, consisting of one machine and of jobs to be processed, as a system to be controlled at discrete instants, using a combination of discrete and continuous controls. At time zero and at each job completion time $\gamma,\dots,\gamma_{n-1}$ thereafter, a control is applied, specifying *which job* the machine will process next, and at what time this processing will start. At the κ–th control point ($\kappa = 1,\dots,n$), the former decision may be represented by the (discrete) assignment variables $u_{j\kappa}$ (equal to one if job j is processed in position κ, and zero otherwise) for all $j \in N$; and the latter decision by the (continuous) positional start time τ_κ. The next, $(\kappa+1)$–st control instant will then be simply the positional completion time γ_κ. These $2n+n^2$ variables (τ,γ,u) must satisfy a number of constraints, described below. Note that, except for the initial control at instant zero, the other control instants $\gamma_1,\dots,\gamma_{n-1}$ are not specified in advance but are implied by earlier decisions, because of the different job processing times and also the possibility of inserting machine idle time between successive jobs. (Recall that idle time between jobs may be unavoidable, e. g., in the presence of release dates.)

The assignment variables must satisfy the *assignment constraints,*

$$\sum_{\kappa\in N} u_{j\kappa} = 1 \qquad \text{for all } j \in N, \tag{5.1}$$

$$\sum_{j\in N} u_{j\kappa} = 1 \qquad \text{for all } \kappa \in N, \tag{5.2}$$

$$u_{j\kappa} \in \{0,1\} \qquad \text{for all } j \in N,\ \kappa \in N, \tag{5.3}$$

where (5.1) requires that each job is processed exactly once; (5.2) that there is exactly one job in each position; and (5.3) are the integrality requirements on the assignment variables. The positional date variables must satisfy

$$\tau_\kappa \geqslant \gamma_{\kappa-1} \qquad \text{for all } \kappa \in N, \tag{5.4}$$

$$\gamma_\kappa = \tau_\kappa + \sum_{j\in N} p_j u_{j\kappa} \qquad \text{for all } \kappa \in N, \tag{5.5}$$

where $\gamma_0 := 0$ for convenience; (5.4) requires that the job in each position cannot begin until the job in the previous position is completed on the machine; and (5.5) defines the positional completion times given the positional start times and the assignment variables. We may also enforce additional constraints, such as for natural release dates r_j and deadlines $\overline{d}_j$,

$$\tau_\kappa \geqslant \sum_{j\in N} r_j u_{j\kappa} \qquad \text{for all } \kappa \in N, \tag{5.6}$$

$$\gamma_\kappa \leqslant \sum_{j\in N} \overline{d}_j u_{j\kappa} \qquad \text{for all } \kappa \in N, \tag{5.7}$$

respectively; as well as corresponding positional constraints,

$$\tau_\kappa \geqslant \rho_\kappa \qquad \text{for all } \kappa \in N, \tag{5.8}$$

$$\gamma_\kappa \leqslant \overline{\delta}_\kappa \qquad \text{for all } \kappa \in N. \tag{5.9}$$

The *positional deadlines* $\overline{\delta}_\kappa$ as well as *positional due dates* δ_κ, may be used to require that a minimum *number* of jobs be completed by these given dates, without specifying the identity of these jobs. Thus a contractual or licensing requirement may specify that at least 5 jobs be completed within a week, and a further 20 jobs within a month. See Hall [Hal86] and Hall et al. [HSS91] for further motivation, description and analysis of scheduling problems with positional (generalized) due dates and deadlines, and for additional references. Similarly, *positional release dates* ρ_κ may restrict the number (but not the identity) of jobs available by given dates. Such constraints may arise, for instance, if a supplier has a limited production capacity for materials or subassemblies. As an example, consider the two–machine flow shop problem of Ahmadi et al. [AADT92], where the first machine is a *batch machine* which can process a batch of any b jobs in t time units. Hoogeveen and van de Velde [HV95a] formulate this as a single machine problem (on the second machine) with positional release dates $\rho_\kappa := \beta t$ for $\kappa = 1+(\beta-1)b$ and $\beta = 1,\dots,\lceil n/b\rceil$, where β is the index of a batch on the first machine. They then use Lagrangian relaxation to obtain lower bounds.

The formulation (5.1) – (5.5) allows a natural representation of positional performance measures. In particular, optimizing any objective function which is linear or piecewise linear convex in the positional start times τ_κ and/or completion times γ_κ leads to a mixed integer linear programming problem. Some nonpositional objective functions can also be formulated as linear or piecewise linear convex functions of the positional variables, for example the makespan γ_n already mentioned. In fact, any *symmetric* function, i. e., any function $f(x_1,\dots,x_n)$ which is invariant under a permutation of its variables $x_1,\dots,x_n$, for which each $x_j := g_j(C_j)$ is a linear (or piecewise linear convex) function of the natural completion times C_j, may be expressed as a (linear or piecewise linear convex) function of the positional variables. This includes such symmetric objectives as the mean or maximum of all job completion times, lateness or tardiness.

Lasserre and Queyranne report on some empirical results with such formulations. For the problems they tried, they find that positional formulations are not dominated by the best time–indexed formulation (see Section 3 above) in terms of linear programming lower bounds, and that they may require far fewer variables. As another example, Dauzère–Pérès [Dau94] uses positional variables to represent the number of late jobs in a one–machine problem with release dates. Solving $O(\log n)$ linear programs yields a lower bound on the minimum number of late jobs, which is independent of a big–M coefficient used in the formulation. The lower bound obtained from this formulation is then used to empirically assess the quality of a heuristic solution method.

Such formulations and results motivate polyhedral studies of models with positional date and/or assignment variables.

5.2 Polyhedral Structure: Positional Date Variables

We now review polyhedral results [LQ92] with positional date variables for single machine problems. As in Section 2.3 above, these results deal with unconstrained scheduling and the corresponding polyhedra are closely related to some supermodular polyhedra, although in an interestingly distinct way. The structure of constrained problems (e. g., with release dates, deadlines, or precedence constraints) has, to the best of the authors' knowledge, not been studied for positional date variables.

The set of positional date vectors (τ,γ) of feasible schedules is the union of $n!$ (translated) polyhedral cones, each corresponding to a permutation schedule. In contrast with Section 2, however, these cones need not be disjoint. In addition, they are not full–dimensional since all feasible (τ,γ) vectors satisfy the equation

$$\sum_{\kappa=1}^{n}(\gamma_\kappa-\tau_\kappa)=\sum_{j\in N}p_j\ .$$

All these cones have the same extreme direction vectors (y^λ,y^λ) where $y^\lambda:=\sum_{\kappa\geqslant\lambda}e^\kappa$ for $\lambda=1,\ldots,n$, corresponding to idle time inserted just before the job in position λ. It then follows from [Bal88] or [QW92] that the convex hull $Q_{\tau,\gamma}$ of the union of these cones is a convex polyhedron with the same extreme rays. To describe this convex hull, assume that the jobs are ranked in SPT order, that is,

$$0<p_1\leqslant p_2\leqslant\cdots\leqslant p_n \tag{5.10}$$

and let

$$g(S):=\sum_{\kappa\leqslant|S|}p_\kappa \qquad \text{for all } S\subseteq N. \tag{5.11}$$

This set function $g:2^N\to\mathbb{R}$ is normalized, increasing, and supermodular.

Theorem 5.1. [LQ92] *The convex hull of the set of all feasible positional date vectors* (τ,γ) *is given by:*

$$\begin{aligned} Q_{\tau,\gamma}:=\{(\tau,\gamma)\in\mathbb{R}^N\times\mathbb{R}^N:\quad \tau_\kappa-\gamma_{\kappa-1} &\geqslant 0 \quad \textit{for } \kappa\in N; &(5.12)\\ \sum_{\kappa\in S}(\gamma_\kappa-\tau_\kappa) &\geqslant g(S) \quad \textit{for } S\subset N\ (S\neq\emptyset); &(5.13)\\ \sum_{\kappa\in N}(\gamma_\kappa-\tau_\kappa) &= g(N)\quad\}\ , &(5.14)\end{aligned}$$

where $\gamma_0:=0$.

We now sketch a proof of this theorem, distinct from that in [LQ92]. First, it is easy to verify that the constraints in (5.12) – (5.14) are valid for all feasible positional date vectors. Now consider the following change of variables,

$$\alpha_\kappa:=\tau_\kappa-\gamma_{\kappa-1} \qquad\text{and}\qquad \beta_\kappa:=\gamma_\kappa-\tau_\kappa \qquad\text{for all } \kappa\in N. \tag{5.15}$$

When (τ,γ) corresponds to a feasible schedule, α_κ is the amount of idle time just before the κ–th job, and β_κ is its processing time. This change of variables is linear and one–to–one, with inverse

$$\tau_\kappa=\alpha_\kappa+\sum_{\lambda<\kappa}(\beta_\lambda+\alpha_\lambda) \qquad\text{and}\qquad \gamma_\kappa=\sum_{\lambda\leqslant\kappa}(\beta_\lambda+\alpha_\lambda). \tag{5.16}$$

Thus the set of all $(\alpha,\beta) \in \mathbb{R}^{N\times N}$ for which (τ,γ) defined by (5.16) are in the polyhedron $Q_{\tau,\gamma}$ defined by (5.12) – (5.14), is the polyhedron

$$Q'_{\alpha,\beta} := \{(\alpha,\beta) \in \mathbb{R}^N \times \mathbb{R}^N : \quad \alpha_\kappa \geqslant 0 \quad \text{for all } \kappa \in N; \tag{5.17}$$

$$\sum_{\kappa\in S} \beta_\kappa \geqslant g(S) \quad \text{for all } S \subset N \ (S \neq \emptyset); \tag{5.18}$$

$$\sum_{\kappa\in N} \beta_\kappa = g(N) \quad \} \ . \tag{5.19}$$

Furthermore, there is a one–to–one correspondence between extreme points (resp., extreme rays) of $Q_{\tau,\gamma}$ and those of $Q'_{\alpha,\beta}$. By (5.17) – (5.19), $Q'_{\alpha,\beta}$ is the Cartesian product $Q'_{\alpha,\beta} = \mathbb{R}^N_+ \times B(g)$ of polyhedra $\mathbb{R}^N_+$ and $B(g)$, where $\mathbb{R}^N_+$ is the nonnegative orthant in the space of the α variables, and $B(g)$ is the base polytope associated with the supermodular system $(2^N, g)$ in the space of the β variables. The extreme rays of $Q'_{\alpha,\beta}$ have thus direction $(e^\kappa, 0)$ where e^κ are the unit vectors in the α space. Its extreme points $(0,\beta^\pi)$ are defined, by the greedy algorithm for base polytopes, using every permutation π of N as follows:

$$\beta^\pi_{\pi(\kappa)} := g(S_\kappa) - g(S_{\kappa-1}) \quad \text{where } S_0 := \emptyset \text{ and } S_\kappa := S_{\kappa-1} \cup \{\pi(\kappa)\} \quad \text{for all } \kappa \in N.$$

Therefore $\beta^\pi_{\pi(\kappa)} = p_\kappa$. Every point of $Q_{\tau,\gamma}$ is the image of a convex combination of these extreme points $(0,\beta^\pi)$ plus any $(\alpha,0) \geqslant 0$. Thus, using (5.16), every point of $Q_{\tau,\gamma}$ is a convex combination of permutation schedules plus some nonnegative idle times. This completes the proof of Theorem 5.1.

It follows from the proof of this theorem that any linear objective

$$\sum_{\kappa\in N} (\varphi_\kappa \tau_\kappa + \psi_\kappa \gamma_\kappa) \ , \tag{5.20}$$

where φ_κ and ψ_κ are arbitrary *positional weights*, can be minimized over the *simple positional date polyhedron* $Q_{\tau,\gamma}$ by a greedy algorithm [LQ92]. This algorithm may be directly derived by equations (5.15) – (5.16) from the greedy algorithm for the Cartesian product $Q'_{\alpha,\beta} = \mathbb{R}^N_+ \times B(g)$; the details are left to the reader. Since this algorithm results in a (positional) schedule, it also follows that the continuous formulation (5.1) – (5.2), (5.4) – (5.5) and $u \geqslant 0$ is sufficient, in the absence of other (release date, deadline or precedence) constraints, to produce a feasible schedule minimizing the weighted positional objective (5.20).

The polyhedral structure of $Q_{\tau,\gamma}$ is also easily described using the combinatorial equivalence to polyhedron $\mathbb{R}^N_+ \times B(g)$ shown above. The argument in this and the next paragraph is much simpler than the proof of the corresponding Theorem 4 in [LQ92]. First, by a result of Shapley [Sha71], the base polytope $B(g) \subseteq \mathbb{R}^N$ is of dimension $n-q$ if and only if q is the maximum number of subsets in a partition $\{S_1,\dots,S_q\}$ of N (in nonempty subsets) such that $\sum_{i=1}^q g(S_i) = g(N)$. Therefore, if $p_1 < p_n$ then $q = 1$ and $\dim Q_{\tau,\gamma} = 2n-1$. Otherwise, $p_1 = p_2 = \cdots = p_n$, the so–called *unit time jobs* case (assuming w. l. o. g. that all $p_j = 1$), and then $q = n$ and $\dim Q_{\tau,\gamma} = n$.

We next determine, for the case $p_1 < p_n$, those inequalities in (5.12) – (5.13) which are facet defining for $Q_{\tau,\gamma}$. When $\dim(B(g)) = n-1$, a proper subset S of N induces a facet of a base polytope $B(g) \subseteq \mathbb{R}^N$ if and only if S is g–inseparable and its complement $N \setminus S$ is $g^\sharp$–inseparable. For the supermodular function g defined in (5.11), this is equivalent to requiring that $|S| = 1$ or $p_1 < p_{|S|}$, and that $|S| = n-1$ or $p_{|S|+1} < p_n$. Recall also that a minimal linear system for the Cartesian product of given polyhedra is obtained by the simple juxtaposition of minimal linear systems, one for each given polyhedron. Thus, when $p_1 < p_n$, the facet defining inequalities for $Q_{\tau,\gamma}$ are all inequalities (5.12), and all those inequalities (5.13) defined by proper subsets S of N satisfying the conditions just described.

Lasserre and Queyranne [LQ92] provide a simple $O(n \log n)$ separation algorithm for the simple positional scheduling polyhedron $Q_{\tau,\gamma}$. They also study the projections of $Q_{\tau,\gamma}$ into the subspaces of the positional start (resp., completion) dates τ (resp., γ) alone. They obtain structural and algorithmic polyhedral results similar to those described above for $Q_{\tau,\gamma}$. These polyhedral results, in particular the fast separation algorithms, may have application for solving more complex (positional) scheduling problems, e. g., with release date, deadline or precedence constraints, by working in spaces of positional dates only.

5.3 Polyhedral Structure: Assignment Variables

We consider the structure of the convex hull of incidence vectors $u \in \mathbb{R}^{N \times N}$ of the feasible assignments for one–machine problems. First recall that, in the absence of other constraints, the assignment equations (5.1) – (5.2) (except for any one of them, which is a linear combination of the $2n-1$ other equations) and all nonnegativity constraints $u_{j\kappa} \geqslant 0$ exactly describe the convex hull of the $0/1$–valued u vectors defining assignments.

We first consider feasible assignments for one–machine scheduling with (natural) release dates r_j and deadlines $\overline{d}_j$. To derive inequalities in the assignment variables $u_{j\kappa}$ sufficient to express the feasibility of a corresponding schedule, consider any two positions λ and μ with $1 \leqslant \lambda \leqslant \mu \leqslant n$. In any feasible schedule, the completion time of the μ–th job, which is at least the release date of the λ–th job plus the processing time of all jobs on positions $\lambda, \ldots, \mu$, must not exceed its deadline:

$$\sum_{j \in N} r_j u_{j\lambda} + \sum_{\kappa=\lambda}^{\mu} \sum_{j \in N} p_j u_{j\kappa} \leqslant \sum_{j \in N} \overline{d}_j u_{j\mu} \tag{5.21}$$

so these are valid inequalities for feasible assignments. A direct argument [LQ92] proves that a vector u satisfying (5.1) – (5.3) defines a feasible schedule if and only if it satisfies these $O(n^2)$ inequalities (5.21). We outline below an alternative argument, showing that these inequalities are, in a restricted sense, the strongest inequalities characterizing feasible schedules in the space of the $u_{j\kappa}$ variables.

First, eliminate the completion time variables γ by direct substitution using the defining equations (5.5). Ignoring the assignment constraints (5.1) – (5.3), we now seek to characterize those vectors $u \in \mathbb{R}^{N \times N}$ for which there exist a vector $\tau \in \mathbb{R}^N$ satisfying

$$\begin{aligned}
\tau_{\kappa-1} - \tau_\kappa + \sum_{j \in N} p_j u_{j,\kappa-1} &\leq 0 \qquad \forall \kappa = 2, \ldots, n && (5.22)\\
-\tau_\kappa + \sum_{j \in N} r_j u_{j\kappa} &\leq 0 \qquad \forall \kappa \in N && (5.23)\\
\tau_\kappa - \sum_{j \in N} \left(\overline{d}_j - p_j\right) u_{j\kappa} &\leq 0 \qquad \forall \kappa \in N && (5.24)
\end{aligned}$$

which correspond to (5.4), (5.6) and (5.7), respectively. Using nonnegative multipliers ξ_κ, η_κ and χ_κ for inequalities (5.22), (5.23) and (5.23), respectively, the condition that each variable τ_κ be eliminated from the resulting nonnegative combination of these inequalities is

$$\xi_{\kappa+1} - \xi_\kappa - \eta_\kappa + \chi_\kappa = 0 \qquad \forall \kappa \in N, \tag{5.25}$$

where $\xi_1 := 0$ and $\xi_{n+1} := 0$ for convenience. Now observe that the so–called *projection cone*, viz, the set of all $(\xi, \eta, \chi) \geqslant 0$ satisfying (5.25), is precisely the set of feasible s–t flows in an uncapacitated network defined as follows. The node set is $N \cup \{s, t\}$, where $s, t \notin N$ are an additional source and sink. The arcs $(\kappa - 1, \kappa)$, (s, κ) and (κ, t), correspond to the multipliers ξ_κ, η_κ and χ_κ, respectively. Feasible flows are nonnegative vectors (ξ,η,χ) satisfying *the balance equations* (5.25). Since this network is acyclic and uncapacitated, every feasible *s–t* flow is a nonnegative combination of flows on *s–t* paths; in other words, the latter define the extreme rays of the projection cone. An *s–t* path $P(\lambda,\mu)$ is defined by any pair $(\lambda,\mu) \in N^2$ where $\lambda \leqslant \mu$, and its (normalized) flow vector $(\xi^{P(\lambda,\mu)}, \eta^{P(\lambda,\mu)}, \chi^{P(\lambda,\mu)})$ has as only nonzero components

$$\eta_\lambda^{P(\lambda,\mu)} = \xi_\kappa^{P(\lambda,\mu)} = \chi_\mu^{P(\lambda,\mu)} = 1 \qquad \text{for all } \kappa \text{ with } \lambda < \kappa \leqslant \mu.$$

These path flow vectors generate precisely the path inequalities (5.21). Thus these inequalities characterize the set of all (otherwise unrestricted) vectors $u \in \mathbb{R}^{N\times N}$ for which there exist (τ,γ) vectors satisfying (5.4) – (5.7).

These inequalities, however, are not strong enough to define the convex hull of incidence vectors of feasible assignments. In fact, such polyhedral results are probably very difficult to obtain since deciding the existence of a feasible schedule is already NP–hard. Thus, determining the dimension of this polyhedron, and a fortiori characterizing its facets, also appear to be difficult – unless one chooses to study an appropriate relaxation, e. g., a monotonization of the polyhedron, or problems with only release dates, or only deadlines. For the latter two cases, existence is easy, dimension is unknown, and optimization/separation are $\mathcal{NP}$–hard.

We now briefly consider feasible assignments for one–machine scheduling with precedence constraints. For general precedence constraints, existence is trivial, dimension is unknown, and optimization/separation are $\mathcal{NP}$–hard. A simple way to express a precedence constraint, say, that job j must precede job k, is

$$\sum_{\kappa=1}^{\lambda} u_{k\kappa} \leqslant \sum_{\kappa=1}^{\lambda-1} u_{j\kappa} \qquad \forall \lambda = 2,\ldots,n, \tag{5.26}$$

requiring that, if job k is in one of the first λ positions, then job j must be in one of the first $\lambda-1$ positions. These inequalities, when applied to $0/1$ assignment vectors, suffice to define feasible assignments. They are not strong enough, however, to define the convex hull of incidence vectors of feasible assignments. Indeed, observe that, even for the *single* precedence constraint that job j be processed before job k, the dimension of the convex hull of incidence vectors of feasible assignments decreases by two, due to the equations $u_{jn} = 0$ and $u_{k1} = 0$, and several types of new facet–defining inequalities arise (for example, $u_{k2} + u_{\ell 1} + u_{\ell 2} \leqslant 1$ where $\ell \neq j,k$). The polyhedral structure of precedence constrained assignments deserves further study.

5.4 Multiple Machine Extensions

It is relatively easy to extend formulations involving positional date and assignment variables to multiple machine problems which clearly involve a single permutation of the jobs, such as the permutation and no–wait flow shop problems discussed below. This approach may also be extended, at the cost of a larger number of variables, to other multiple machine problems.

In a *flow shop*, the machines are numbered 1 to m, and each job must visit all the machines in this order. In a *permutation flow shop*, each machine must process all the jobs in the same order, to be determined. Although the latter requirement may result in a loss of optimality for certain criteria (e. g., for the makespan when $m \geqslant 4$), it often follows from a physical constraint determined by the materials management system, or from a managerial constraint aiming to maintain a smooth flow of materials in the shop. Thus a single permutation of jobs is sought, and may be formulated using assignment variables $u_{j\kappa}$ as before, and positional start dates $\tau_{i\kappa}$ and completion times $\gamma_{i\kappa}$ for each position $\kappa = 1,\ldots,n$ and each machine $i = 1\ldots,m$. These date variables are restricted by straightforward extensions of the conditions (5.4) – (5.5) to each machine i, namely,

$$\tau_{i\kappa} \geqslant \gamma_{i,\kappa-1} \qquad \text{for all } \kappa \in N, i \in M, \tag{5.27}$$

$$\gamma_{i\kappa} = \tau_{i\kappa} + \sum_{j\in N} p_{ij} u_{j\kappa} \qquad \text{for all } \kappa \in N,\ i \in M, \tag{5.28}$$

where $M := \{1,\ldots,m\}$ denotes the set of machines, p_{ij} is the processing time of job j on machine i, and all $\gamma_{i0} := 0$. In addition to these machine requirements, the positional dates must satisfy the constraints that the processing of a job be completed on a machine before it can be moved to the next machine,

$$\tau_{i\kappa} \ \geqslant \ \gamma_{i-1,\kappa} \qquad \text{for all } \kappa \in N, \text{ for all } i \in M, \tag{5.29}$$

where all $\gamma_{0j} := 0$ (or γ_{0j} may denote the release date of job j). A *no–wait flow shop* is a permutation flow shop where no job is allowed to wait between its processing on successive machines. This is simply enforced by replacing the inequality in constraints (5.29) by an equality. These equalities may in turn be used to eliminate, for each job, all date variables except, say, the start times on the first machine. Exploratory empirical experiments by the first author on some randomly generated three machine permutation flowshop problems with makespan objective suggest that such positional formulations may lead to good lower bounds.

It is interesting to relate the formulation (5.27) – (5.29) above to an integer programming model proposed by Wagner [Wag59] in 1959 for the 3–machine flow shop problem with makespan objective. Wagner uses the same assignment variables as above. Instead of positional date variables, however, he uses what we may now call *positional idle time variables*, which we may define as follows

$$\sigma_{i,\kappa-1} := \tau_{i\kappa} - \gamma_{i,\kappa-1} \qquad \text{and} \qquad \upsilon_{i-1,\kappa} := \tau_{i\kappa} - \gamma_{i-1,\kappa} \tag{5.30}$$

for all $i = 2,3$ and $\kappa = 2,\ldots,n$. Thus $\sigma_{i,\kappa-1}$ may be interpreted as the slack variable for constraint (5.27), i. e., as the idle time on machine i between the jobs in positions $\kappa-1$ and κ; and $\upsilon_{i-1,\kappa}$ as the slack variable for constraint (5.29), i. e., as the idle time for the job in position κ between machine $i-1$ and i. To the assignment constraints (5.1) – (5.3), Wagner adds $2(n-1)$ constraints

$$\sum_{j\in N} p_{2j} u_{j\kappa} - \sum_{j\in N} p_{1j} u_{j,\kappa+1} + \sigma_{2\kappa} + \upsilon_{1\kappa} - \upsilon_{1,\kappa+1} = 0 \tag{5.31}$$

$$\sum_{j\in N} p_{3j} u_{j\kappa} - \sum_{j\in N} p_{2j} u_{j,\kappa+1} - \sigma_{2\kappa} + \sigma_{3\kappa} + \upsilon_{2\kappa} - \upsilon_{2,\kappa+1} = 0 \tag{5.32}$$

for $\kappa = 1,\ldots,n-1$. See [GW64] for early experimental results with Wagner's formulation. We now compare Wagner's formulation to that using positional dates and assignment variables described above. Using (5.30), these equations simplify to

$$\Big(\gamma_{1,\kappa+1} - \gamma_{1\kappa} - \sum_{j\in N} p_{1j} u_{j,\kappa+1}\Big) - \Big(\gamma_{2\kappa} - \tau_{2\kappa} - \sum_{j\in N} p_{2j} u_{j\kappa}\Big) = 0 \tag{5.33}$$

$$\Big(\gamma_{2,\kappa+1} - \tau_{2,\kappa+1} - \sum_{j\in N} p_{2j} u_{j,\kappa+1}\Big) - \Big(\gamma_{3\kappa} - \tau_{3\kappa} - \sum_{j\in N} p_{3j} u_{j\kappa}\Big) = 0 \tag{5.34}$$

for $\kappa = 1,\ldots,n-1$. Equation (5.34) is implied by (5.28). Equation (5.33) is also implied by (5.28) and by the equations $\tau_{1,\kappa+1} = \gamma_{1\kappa}$ requiring no idle time on machine 1. Thus the continuous relaxation of formulation (5.1) – (5.3) and (5.27) – (5.29) with (5.27) as equalities for the first machine, is at least as strong as Wagner's formulation.

Dauzère–Pérès and Lasserre [DL95] consider the general flow shop problem, without the "permutation" restriction that every machine process all jobs in the same order. They propose a formulation with m^2n^2 assignment variables and m^2n "event" positional date variables. Let $O = M \times N$ denote the set of all mn operations $o = (i, j)$. Let $O(i) = \{o = (i, j) : j \in N\}$ the set of operations to be performed on machine i. Define the assignment variables $u_{o\kappa} = 1$ if operation o is the κ–th operation overall, and zero otherwise, for $o \in O$ and $\kappa \in K = \{1, \ldots, nm\}$. These assignment variables must satisfy the usual assignment constraints

$$\sum_{\kappa \in K} u_{o\kappa} = 1 \qquad \text{for all } o \in O, \tag{5.35}$$

$$\sum_{o \in O} u_{o\kappa} = 1 \qquad \text{for all } \kappa \in K, \tag{5.36}$$

$$u_{o\kappa} \in \{0, 1\} \qquad \text{for all } o \in O,\ \kappa \in K, \tag{5.37}$$

as well as the *operations precedence constraints*

$$\sum_{\lambda \leqslant \kappa} u_{o,\lambda} \leqslant \sum_{\lambda \leqslant \kappa - 1} u_{s(o),\lambda} \qquad \text{for all } o \in O \setminus O(1),\ \kappa \in K \setminus \{2\}. \tag{5.38}$$

where $s(o) = (i-1, j)$ is the operation that precedes operation $o = (i, j)$ in the routing for the same job j. This last set of constraint forces operation o to a position after that of every preceding operation for the same job.

Let θ^i_κ denote the date of the κ–th *event* on machine i, where θ^i_κ is the start time of the κ–th operation if this operation is on machine i itself, and can be any time between the completion time of the ι–th operation and the start time of the λ–th operation, otherwise, where ι (resp., λ) is the position of the latest (resp., earliest) operation on machine i before (resp., after) position κ. These event positional dates must satisfy the *initial conditions*

$$\theta^i_1 \geqslant 0 \qquad \text{for all } i \in M \tag{5.39}$$

and the *dynamics constraints*

$$\theta^i_{\kappa+1} \geqslant \theta^i_\kappa + \textstyle\sum_{o \in O(i)} p_o u_{o\kappa} \qquad \text{for all } i \in M,\ \kappa \in K. \tag{5.40}$$

If the κ–th event is the start of an operation on machine i, these dynamics constraints force the next event on machine i to wait for the completion of this operation; otherwise, they simply require that the next event be no earlier than the current event. The event positional dates must also satisfy the *machine precedence constraints*

$$\theta^i_\kappa \geqslant \theta^{i-1}_\kappa \qquad \text{for all } i \in M \setminus \{1\},\ \kappa \in K. \tag{5.41}$$

These constraints force the events with same position to be no earlier on machine i than on any preceding machine.

Dauzère–Pérès and Lasserre [DL95] show that the constraints (5.35)–(5.41) define a correct formulation for the general flow shop problem. This formulation is easily extended to encompass (natural and positional) release dates and deadlines, as in (5.6) to (5.9). These release dates and deadlines may be associated with jobs, or even with individual operations. Dauzère–Pérès and Lasserre also prove a flow shop analogue of the path inequalities (5.21) for the case of natural job release dates and deadlines.

As in Section 5.1, the positional flow shop formulations presented above can accomodate positional performance measures, such as the makespan, and symmetric measures such as mean or maximum job (or operation) completion time, lateness or tardiness.

Assignment and positional variables may extend to other multiple machine situations, possibly at the cost of a further increase in the number of variables. For a simple example with unrelated parallel machines, Bruno et al. [BCS74] express the total completion time as

$$\sum_{j\in N} C_j = \sum_{j\in N}\sum_{i}\sum_{\omega} \omega p^i_j u^i_{j\omega} \tag{5.42}$$

where the assignment variables are now $u^i_{j\omega}$, equal to one if job j is in ω–th position *from the end* on machine i. As in Section 5.1, similar formulations obtain for other symmetric objectives, such as makespan, and mean or total value of job lateness or tardiness.

6 TSP Variables

> "The traveling salesman problem, in its symmetric as well as its asymmetric version, is the classical model for the optimal sequencing of items in all those situations where the cost of a sequence can be expressed as the sum of costs of successive pairs of items" (Balas, Fischetti, and Pulleyblank [BFP95]).

There is a great variety of problems that can be formulated by this model. We will focus on the non-preemptive single machine problem with sequence–dependent processing times. Whereas the disjunctive formulation based on natural date variables we presented in Section 2 might be more appropriate for minimizing the weighted sum of completion times, it seems to be adequate to use traveling salesman problem variables for minimizing the overall completion time, i. e., the makespan. The facial structure of the symmetric as well as the asymmetric traveling salesman polytope has been the object of considerable research. We refer the reader to the survey of properties of both polyhedra due to Grötschel and Padberg [GP85], and to Balas and Fischetti [BF93] and Queyranne and Wang [QW93] for the recent developments with regard to the asymmetric and the symmetric case, respectively.

6.1 Minimizing the Makespan Using the TSP

The sequencing problem we are interested in is defined as follows. We are given a set $N = \{1,\dots,n\}$ of jobs to be nonpreemptively processed on a single disjunctive machine. Each job j has a processing time p_j, and there are changeover or setup times s_{jk} that arise if job j is sequenced immediately before k. We want to minimize the maximal completion time. It is straightforward to model this problem as an asymmetric TSP since the maximal completion time of a sequence depends only on its imposed sum of changeover times. Let $G_{n+1} = (N_0, A_{n+1})$ denote the complete directed graph on $N_0 := N \cup \{0\}$ where node 0 is a designated home city, i. e. every tour starts and finishes in 0. From the scheduling point of view, 0 is a dummy job scheduled twice, once at the beginning, once at the end of the sequence. We set $p_0 := 0$ and $s_{j0} := 0$ for $j \in N$, whereas $s_{0j} \geqslant 0$ may be used as a kind of release date. Each sequence of the jobs can be seen as a tour, i. e., a Hamiltonian circuit in G_{n+1} and vice versa. If we assign to each arc $(j,k) \in A_{n+1}$ the sequence–dependent processing time p_{jk} of job j, $p_{jk} := p_j + s_{jk}$, as its cost, a tour of minimum total cost corresponds to a sequence minimizing the makespan. The asymmetric traveling salesman (ATS) polytope is the convex hull of incidence vectors of all tours, i. e., it is the convex hull of all $z \in \{0,1\}^{A_{n+1}}$ satisfying the *degree constraints*

$$\sum_{k\in N_0\setminus\{j\}} z_{jk} = \sum_{k\in N_0\setminus\{j\}} z_{kj} = 1 \qquad \text{for all } j \in N_0$$

and the *subtour elimination constraints*

$$z(S, N_0 \setminus S) = \sum_{j \in S, k \in N_0 \setminus S} z_{jk} \geqslant 1 \quad \text{for all } \emptyset \subset S \subset N_0,$$

that are written down here in its cut form. Thus, minimizing the makespan could be done by solving a linear programming problem over the asymmetric traveling salesman polytope, if we knew a complete description.

6.2 Precedence Constraints

If there are restrictions on the sequences in terms of given precedences between pairs of jobs, we obtain the *precedence constrained asymmetric traveling salesman polytope* (cf., [BFP95]) and the *sequential ordering problem* (cf., e. g., [AEGS93]). The sequential ordering problem is to determine in a complete directed graph with arc costs a Hamiltonian path that does not violate the precedences and has minimum total cost. This problem has been attacked with a cutting plane approach by Ascheuer, Escudero, Grötschel, and Stoer [AEGS90, AEGS93].

We call a tour in G_{n+1} feasible if it satisfies the precedence constraints. Because of these constraints there are several arcs in A_{n+1} that cannot be used by any feasible tour. Balas, Fischetti and Pulleyblank [BFP95] derived the following complete characterization of removable arcs, the interesting direction of which can be proved by a simple shrinking argument.

Proposition 6.1. [BFP95] *An arc $(j,k) \in A_{n+1}$ cannot be used by any tour if and only if it satisfies one of the following conditions:*

(a) $j = 0$ *and there exists* $\ell \in N_0 \setminus \{k\}$ *such that* $(\ell, k) \in A$*;*

(b) $k = 0$ *and there exists* $\ell \in N_0 \setminus \{j\}$ *such that* $(j, \ell) \in A$*;*

(c) $k \to j$*;*

(d) *there exists* $\ell \in N_0 \setminus \{j,k\}$ *such that* $(j,\ell), (\ell,k) \in A$.

This proposition allows us to delete some arcs. Let $G^D_{n+1} := (N_0, A^D_{n+1})$ be the digraph obtained from G_{n+1} by removing all the arcs for which one of the four conditions stated in Proposition 6.1 holds. For the remainder of this section we assume to work with G^D_{n+1}. That implies in particular that the precedence constrained asymmetric traveling salesman (PCATS) polytope is contained in $\mathbb{R}^{A^D_{n+1}}$.

Balas, Fischetti and Pulleyblank [BFP95] presented essentially four different families of valid inequalities for the precedence constrained asymmetric traveling salesman polytope. Three of them are strengthenings of the subtour elimination constraints and the fourth contains inequalities that may also be seen in this way. Balas et al. discuss various equivalent forms of these inequalities and show that they dominate other inequalities proposed earlier in the literature.

Let I be an ideal of D. In each feasible tour there has to be a maximal element of I the successor of which in this tour cannot be in I. Therefore each feasible tour satisfies the *predecessor inequalities*

$$z(\max(I), N_0 \setminus I) \geqslant 1, \tag{6.1}$$

where I is an ideal, and $\max(I)$ are the maximal elements of I. By a similar argument one obtains the *successor inequalities*

$$z(N_0 \setminus F, \min(F)) \geqslant 1,$$

where F is a filter of D, and $\min(F)$ are its minimal elements. Balas, Fischetti and Pulleyblank show under some technical conditions that both classes define facets if A is both, an ideal as well as a filter.

If $(j,k) \in A$, any feasible tour contains a path from j to k. This path cannot use any node that has to precede j, that has to succeed k, or that is identical to 0. As we did for the series inequalities in the natural date variables (cf., Section 2.4.3) we can even replace j and k by job subsets $J,K \subset N$ such that every job of J has to precede each job of K. Let $A \subseteq N_0$ be such that $J \subseteq A$ and $K \subseteq (N_0 \setminus A)$. Furthermore, define $L := \mathrm{Pred}(J) \cup \mathrm{Succ}(K) \cup \{0\}$. Then every feasible tour satisfies the following *predecessor–successor inequality*,

$$z(A \setminus L, N_0 \setminus (A \cup L)) \geqslant 1.$$

The fourth family of Balas, Fischetti and Pulleyblank are the *precedence cycle breaking inequalities*. Let $S_1,\dots,S_q \subset N$ be $q \geqslant 2$ disjoint job sets, and assume that $\mathrm{Succ}(S_1) \cap S_2 \neq \emptyset, \dots, \mathrm{Succ}(S_{q-1}) \cap S_q \neq \emptyset, \mathrm{Succ}(S_q) \cap S_1 \neq \emptyset$. Now, suppose there exists a feasible tour satisfying all the subtour elimination constraints $z(S_\ell, N_0 \setminus S_\ell) \geqslant 1$ with equality. Then the nodes of each S_ℓ are visited consecutively, i. e. they build a subpath since each set S_ℓ is entered and left exactly once. By shrinking the sets S_ℓ and using the assumption we see that the (shrinked) tour would satisfy a cycle in the (shrinked) precedences, a contradiction. Therefore, the precedence cycle breaking inequalities

$$\sum_{\ell=1}^{q} z(S_\ell, N_0 \setminus S_\ell) \geqslant q+1$$

are valid for the PCATS polytope.

We close this section on the precedence–constrained asymmetric traveling salesman polytope by mentioning that Balas, Fischetti and Pulleyblank also provide a lifting procedure that generalizes the one of Balas and Fischetti [BF93] for the pure asymmetric traveling salesman polytope.

6.3 TSP with Time Windows

Other typical restrictions arising in scheduling are release dates and deadlines. Assume that the triangle inequalities are satisfied, i. e., $s_{j\ell} \leqslant s_{jk} + p_k + s_{k\ell}$, and that each job j is released at time r_j and must start before time $\overline{d}'$. The interval $[r_j, \overline{d}'_j]$ defines a *time window* within which the execution of job j has to start. Including these time constraints into the TSP model leads to the following "big–M formulation" of the asymmetric traveling salesman problem with time windows (cf., e. g., [DLSS88]):

$$\begin{aligned}
\textstyle\sum_{k \in N_0 \setminus \{j\}} z_{jk} &= 1 && j \in N_0, && (6.2)\\
\textstyle\sum_{j \in N_0 \setminus \{k\}} z_{jk} &= 1 && k \in N_0, && (6.3)\\
S_j + p_j + s_{jk} - (1 - z_{jk}) \cdot M_{jk} &\leqslant S_k && j,k \in N, j \neq k, && (6.4)\\
S_j &\leqslant \overline{d}'_j && j \in N, && (6.5)\\
S_j &\geqslant r_j && j \in N, && (6.6)\\
z_{jk} &\in \{0,1\} && j,k \in N_0, j \neq k. && (6.7)
\end{aligned}$$

Here, S_j stands for the start time of job $j \in N$. Whereas (6.2) and (6.3) are the usual degree constraints, (6.5) and (6.6) model the time windows. If the big–M coefficients satisfy $M_{jk} \geqslant \overline{d}'_j + p_j + s_{jk} - r_k$, then the inequalities (6.4) enforce the disjunctive constraints

$$S_k \geqslant S_j + p_j + s_{jk} \qquad \text{or} \qquad S_j \geqslant S_k + p_k + s_{kj}$$

and link the TSP and the date variables. Notice that constraints (6.4) also prevent subtours. As it is often the case, the big–M formulation performs poorly in practice. Ascheuer [Asc95] compared branch&cut algorithms based on the big–M formulation and on a formulation involving TSP variables only. His experiments with instances involving up to 50 nodes (jobs) show that the latter is always preferable, whereby constraints (6.4) – (6.6) are replaced by the subtour elimination constraints and (lifted) *infeasible path inequalities*

$$z(A) \leqslant |A| - 1 \qquad \text{if } A \text{ is the arc set of an infeasible path.} \tag{6.8}$$

However, this model has the disadvantage that it can only be used for minimizing objective functions involving only the TSP variables. Notice, that with the choice $c_{jk} := s_{jk}$ we do not longer minimize the overall completion time since the release dates may enforce idle time. Even in the model using TSP and starting time variables the makespan is a linear objective only if we introduce a start time variable S_0 for the dummy job, and if we add the constraints

$$S_j + p_{j0} - (1 - z_{j0}) \cdot M_{j0} \leqslant S_0, \qquad j \in N.$$

Minimizing the sum of starting times, $\sum_{j \in N} S_j$ or, equivalently, the sum of waiting times, $\sum_{j \in N}(S_j - r_j)$ is known as the *traveling repairman problem* (cf., e. g., [ACP+86]).

Van Eijl [Eij95] suggested to use another type of date variables besides the TSP variables in order to avoid the big–M coefficients. She extracted her model, that can be used for all three types of objectives mentioned above, from one of Maffioli and Sciomachen [MS93] that had even more variables. For $j, k \in N$ define $S_{jk} := S_j \cdot z_{jk}$, i. e.,

$$S_{jk} = \begin{cases} \text{start time of job } j \ , & \text{if } z_{jk} = 1, \\ 0 \ , & \text{otherwise.} \end{cases}$$

Observe that the starting time of job j can be obtained as $S_j = \sum_k S_{jk}$. The mixed–integer model now looks as follows. In addition to the degree constraints (6.2), (6.3), and the integrality constraints (6.7) there are the following constraints:

$$\begin{aligned}
\textstyle\sum_{j \in N_0 \setminus \{k\}} (S_{jk} + (p_j + s_{jk}) z_{jk}) &\leqslant \textstyle\sum_{\ell \in N_0 \setminus \{k\}} S_{k\ell} && k \in N_0, && (6.9) \\
S_{jk} &\leqslant \overline{d}'_j \cdot z_{jk} && j, k \in N_0, j \neq k, && (6.10) \\
S_{jk} &\geqslant r_j \cdot z_{jk} && j, k \in N_0, j \neq k, && (6.11) \\
S_{jk} &\geqslant 0 && j, k \in N_0, j \neq k. && (6.12)
\end{aligned}$$

Notice that inequalities (6.9) not only model the disjunctive constraints but also prevent subtours.

The traveling salesman problem with time windows (TSPTW) is perhaps the right place to mention a technique that is essential for practical problem solving on the base of (mixed) integer programs: preprocessing. We refer the reader to the textbook of Nemhauser and Wolsey [NW88] for a general treatment of this important topic. In the case of the TSPTW, preprocessing can be performed by tightening the time windows, deriving enforced precedences, and elimination of variables (cf., e. g., [DDSS94]). For example, $r_k + p_k > \overline{d}'_j$ for two different jobs j, k implies the precedence (j, k). If the sequence–dependent processing times also fulfil the triangle condition, then we may strengthen this by including the precedence already if $r_k + p_k + s_{kj} > \overline{d}'_j$. The application of Proposition 6.1 then leads immediately to the elimination of variables. Notice also that the polytope associated with the TSPTW is contained in the PCATS polytope with precedences obtained from this preprocessing. Thus, one can use all inequalities that are valid for the PCATS polytope also for the TSPTW polytope.

Additional Notes and References

Three remarks concerning the TSPTW have to be made. First, it should be clear that minimizing the total sum of arc costs subject to time windows is NP–hard, since this is already the case for the TSP. Tsitsiklis [Tsi92] showed that this remains true in case of the traveling repairman. Even worse, Savelsbergh [Sav85] proved that already finding a feasible solution is strongly NP–hard. Note that this also follows from an earlier result of Garey and Johnson [GJ77] who showed that finding a feasible solution to the nonpreemptive single machine scheduling problem with release dates and deadlines is strongly NP–hard, even if the processing times are independent of the sequence.

Whereas we considered the TSP (with time windows) mostly from the scheduling point of view it should be mentioned that one of its main applications is *vehicle routing*. Here, the dummy node 0 is interpreted as the depot for the vehicles where each tour starts and finishes, S_j is the arrival time at node j, and the time windows model the interval between the earliest and the latest service time. For more details we refer the reader to the collection of studies in vehicle routing edited by Golden and Assad [GAe88].

For another single machine problem, involving due dates instead of deadlines that can also be modeled by use of the TSP the reader is referred to Picard and Queyranne [PQ78] (see also [Sha93]).

Acknowledgements

The authors would like to thank Rolf H. Möhring who had the idea to write a survey of polyhedral approaches to machine scheduling, for his encouragements to complete this work. The authors are also grateful to Marjan van den Akker, Norbert Ascheuer, Matteo Fischetti, and three anonymous referees for helpful comments.

References

[AADT92] J. H. Ahmadi, R. H. Ahmadi, S. Dasu, and C. S. Tang. Batching and scheduling jobs with batch and discrete processors. *Operations Research*, 40:750 – 763, 1992.

[AC91] D. Applegate and W. Cook. A computational study of the job–shop scheduling problem. *ORSA Journal on Computing*, 3:149 – 156, 1991.

[ACP+86] F. Afrati, S. Cosmadakis, C. H. Papadimitriou, G. Papageorgiou, and N. Papakonstantinou. The complexity of the traveling repairman problem. *Theor. Info. Appl.*, 20:79 – 87, 1986.

[AEGS90] N. Ascheuer, L. Escudero, M. Grötschel, and M. Stoer. On identifying in polynomial time violated subtour elimination and precedence forcing constraints for the sequential ordering problem. In R. Kannan and W. R. Pulleyblank, editors, *Integer Programming and Combinatorial Optimization*, pages 19 – 28. University of Waterloo, Waterloo, Ontario, 1990. Proceedings of the 1st International IPCO Conference.

[AEGS93] N. Ascheuer, L. Escudero, M. Grötschel, and M. Stoer. A cutting plane approach to the sequential ordering problem (with applications to job scheduling in manufacturing). *SIAM Journal on Optimization*, 3:25 – 42, 1993.

[AFS90] A. von Arnim, U. Faigle, and R. Schrader. The permutahedron of series–parallel posets. *Discrete Applied Mathematics*, 28:3 – 9, 1990.

[AHS93] J. M. van den Akker, C. P. M. van Hoesel, and M. W. P. Savelsbergh. Facet inducing inequalities for single–machine scheduling problems. Memorandum COSOR 93–27, Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands, 1993.

[AHV] J. M. van den Akker, J. A. Hoogeveen, and S. L. van de Velde. Parallel machine scheduling by column generation. Also appeared as working paper COSOR 95–35, Eindhoven University of Technology.

[Akk94] J. M. van den Akker. *LP–Based Solution Methods for Single–Machine Scheduling Problems.* PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 1994.

[AS93] A. von Arnim and R. Schrader. The permutahedron of P_4–sparse posets. Report 93.143, Institute of Computer Science, University of Cologne, Cologne, Germany, 1993.

[AS94] A. von Arnim and A. S. Schulz. Facets of the generalized permutahedron of a poset. Preprint 386/1994, Department of Mathematics, Technical University of Berlin, Berlin, Germany, 1994.

[Asc95] N. Ascheuer. *Hamiltonian Path Problems in the On–Line Optimization of Flexible Manufacturing Systems*. PhD thesis, Technical University of Berlin, Berlin, Germany, 1995.

[Bal69] E. Balas. Machine sequencing via disjunctive graphs: An implicit enumeration algorithm. *Operations Research*, 17:941 – 957, 1969.

[Bal75] E. Balas. A linear characterization of permutation vectors. Management Science Research Report 364, Carnegie Mellon University, Pittsburgh, USA, 1975.

[Bal85] E. Balas. On the facial structure of scheduling polyhedra. *Mathematical Programming Study*, 24:179 – 218, 1985.

[Bal88] E. Balas. On the convex hull of the union of certain polyhedra. *Operations Research Letters*, 7:279 – 283, 1988.

[BCS74] J. L. Bruno, E. G. Coffman Jr., and R. Sethi. Scheduling independent tasks to reduce mean finishing time. *Communications of the Association for Computing Machinery*, 17:382 – 387, 1974.

[Ben71] J. P. Benzécri. Sur l'analyse des préférences. In M. Barbut, editor, *Ordres Totaux Finis*. Paris, 1971.

[BF93] E. Balas and M. Fischetti. A lifting procedure for the asymmetric traveling salesman polytope and a large new class of facets. *Mathematical Programming*, 58:325 – 352, 1993.

[BFP95] E. Balas, M. Fischetti, and W. R. Pulleyblank. The precedence–constrained asymmetric traveling salesman polytope. *Mathematical Programming*, 68:241 – 265, 1995.

[BGT92] P. P. Bhattacharya, L. Georgiadis, and P. Tsoucas. Extended polymatroids: Properties and optimization. In E. Balas, G. Cornuéjols, and R. Kannan, editors, *Integer Programming and Combinatorial Optimization*, pages 298 – 315. Carnegie Mellon University, 1992. Proceedings of the 2nd IPCO Conference.

[BJN+94] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch–and–price: Column generation for solving huge integer programs. In J. R. Birge and K. G. Murty, editors, *Mathematical Programming: State of the Art 1994*, pages 186 – 207. The University of Michigan, 1994.

[BNM93] D. Bertsimas and J. Niño-Moras. Conservation laws, extended polymatroids and multi–armed bandit problems: A unified approach to indexable systems. In G. Rinaldi and L. A. Wolsey, editors, *Integer Programming and Combinatorial Optimization*, pages 355 – 384, 1993. Proceedings of the 3rd IPCO Conference.

[CMM67] R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of Scheduling*. Addison–Wesley, 1967.

[CMM88] M. W. Carter, M. J. Magazine, and T.–S. Moon. A warning about cyclic lot scheduling with sequence dependent set–up times. *International Journal of Production Research*, 26:1281 – 1283, 1988.

[CMSL96] L. M. A. Chan, A. Muriel, and D. Simchi-Levi. Parallel machine scheduling, linear programming and list scheduling heuristics. Department of Industrial Engineering and Management Sciences, Northwestern University, 1996.

[CP95] Z.-L. Chen and W. B. Powell. Solving parallel machine scheduling problems by column generation, 1995. Dept. of Civil Engineering & Operations Research, Princeton University.

[CPS+96] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In F. Meyer auf der Heide and B. Monien, editors, *Automata, Languages and Programming*, number 1099 in Lecture Notes in Computer Science. Springer, Berlin, 1996. Proceedings of the 23rd International Colloquium (ICALP'96).

[CS95] Y. Crama and F. C. R. Spieksma. Scheduling jobs of equal length: Complexity, facets and computational results. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, number 920 in Lecture Notes in Computer Science, pages 277 – 291. Springer, Berlin, 1995. Proceedings of the 4th International IPCO Conference.

[Dau94] S. Dauzère–Pérès. Minimizing late jobs in the general one machine scheduling problem. Technical report, Laboratory for Manufacturing and Productivity, MIT, Cambridge, MA, 1994.

[DDSS94] J. Desrosiers, Y. Dumas, M. M. Solomon, and F. Soumis. Time constrained routing and scheduling. In M. E. Ball, T. L. Magnanti, C. Monma, and G. L. Nemhauser, editors, *Networks*, volume ? of *Handbooks in Operations Research and Management Science*. North–Holland, 1994. to appear.

[DL95] S. Dauzère–Pérès and J.–B. Lasserre. A new mixed–integer formulation of the flow–shop sequencing problem. Paper presented at the Second Workshop on Models and Algorithms for Planning and Scheduling Problems, Wernigerode, Germany, May 1995.

[DLSS88] M. Desrochers, J. K. Lenstra, M. W. P. Savelsbergh, and F. Soumis. Vehicle routing with time windows: Optimization and approximation. In B. L. Golden and A. A. Assad, editors, *Vehicle Routing: Methods and Studies*, pages 65 – 84. North–Holland, Amsterdam, 1988.

[DW60] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101 – 111, 1960.

[DW90] M. E. Dyer and L. A. Wolsey. Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics*, 26:255 – 270, 1990.

[Eij95] C. van Eijl. A polyhedral approach to the delivery man problem. Memorandum COSOR 95–19, Department of Mathematics and Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands, 1995.

[Emm87] H. Emmons. Scheduling to a common due date on parallel uniform processors. *Naval Research Logistics*, 34:803 – 810, 1987.

[FG86] A. Federgruen and H. Groenevelt. The greedy procedure for resource allocation problems: Necessary and sufficient conditions for optimality. *Operations Research*, 34:909 – 918, 1986.

[FG88a] A. Federgruen and H. Groenevelt. Characterization and optimization of achievable performance in queueing systems. *Operations Research*, 36:733 – 741, 1988.

[FG88b] A. Federgruen and H. Groenevelt. $M/G/c$ queueing systems with multiple customer classes: Characterization and control of achievable performance under nonpreemptive priority rules. *Management Science*, 34:1121 – 1138, 1988.

[Fis92] P. C. Fishburn. Induced binary probabilities and the linear ordering polytope: A status report. *Mathematical Social Sciences*, 23:67 – 80, 1992.

[Fuj84] S. Fujishige. Submodular systems and related topics. *Mathematical Programming Study*, 22:113 – 131, 1984.

[Fuj91] S. Fujishige. *Submodular Functions and Optimization*, volume 47 of *Annals of Discrete Mathematics*. North–Holland, Amsterdam, 1991.

[Ful70] D. R. Fulkerson. Blocking polyhedra. In B. Harris, editor, *Graph Theory and Its Applications*, pages 93 – 112. Academic Press, New York, 1970.

[Ful71] D. R. Fulkerson. Blocking and antiblocking pairs of polyhedra. *Mathematical Programming*, 1:168 – 194, 1971.

[GAe88] B. L. Golden and A. A. Assad (eds.). Vehicle Routing: Methods and Studies. North–Holland, Amsterdam, 1988.

[GG77] P. Gaiha and S. K. Gupta. Adjacent vertices on a permutohedron. *SIAM Journal of Applied Mathematics*, 32:323 – 327, 1977.

[GJ77] M. R. Garey and D. S. Johnson. Two–processor scheduling with start–times and deadlines. *SIAM Journal on Computing*, 6:416 – 426, 1977.

[GJR85] M. Grötschel, M. Jünger, and G. Reinelt. Facets of the linear ordering polytope. *Mathematical Programming*, 33:43 – 60, 1985.

[GLLRK79] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287 – 326, 1979.

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, Berlin, 1988.

[Goe96] M. X. Goemans. A supermodular relaxation for scheduling with release dates. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, number 1084 in Lecture Notes in Computer Science, pages 288 – 300. Springer, Berlin, 1996. Proceedings of the 5th International IPCO Conference.

[Goe97] M. X. Goemans. Improved approximation algorithms for scheduling with release dates. Department of Mathematics, MIT, Cambridge, MA; to appear in the proceedings of the 8th ACM–SIAM Symposium on Discrete Algorithms, 1997.

[GP85] M. Grötschel and M. W. Padberg. Polyhedral theory. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, chapter 8, pages 251 – 305. John Wiley & Sons, Chichester, 1985.

[GR71] G. Th. Guilbaud and P. Rosenstiehl. Analyse algébrique d'un scrutin. In M. Barbut, editor, *Ordres Totaux Finis*. Paris, 1971.

[Gra66] R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Tech. J.*, 45:1563 – 1581, 1966.

[GW64] R. J. Giglio and H. M. Wagner. Approximate solutions to the three–machine scheduling problem. *Operations Research*, 12:305 – 324, 1964.

[Hal86] N. G. Hall. Scheduling problems with generalized due date. *IIE Transactions*, 18:220 – 222, 1986.

[HJP75] P. L. Hammer, E. L. Johnson, and U. N. Peled. Facets of regular 0–1–polytopes. *Mathematical Programming*, 8:179 – 206, 1975.

[HSS91] N. G. Hall, S. P. Sethi, and C. Sriskandarajah. On the complexity of generalized due date scheduling problems. *European Journal of Operational Research*, 51:100 – 109, 1991.

[HSSW96] L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off–line and on–line approximation algorithms. Preprint 516/1996, Department of Mathematics, University of Technology, Berlin, Germany, 1996. submitted. Available from ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-516-1996.ps.Z.

[HSW96] L. A. Hall, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off–line and on–line algorithms. In *Proceedings of the 7th ACM–SIAM Symposium on Discrete Algorithms*, pages 142 – 151, 1996.

[HV95a] J. A. Hoogeveen and S. L. van de Velde. Formulating a scheduling problem with almost identical jobs by using positional completion times. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, number 920 in Lecture Notes in Computer Science, pages 292 – 306. Springer, Berlin, 1995. Proceedings of the 4th International IPCO Conference.

[HV95b] J. A. Hoogeveen and S. L. van de Velde. Stronger Lagrangian bounds by use of slack variables: Applications to machine scheduling problems. *Mathematical Programming*, 70:173 – 190, 1995.

[KK86] T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow–time problem. *SIAM Journal on Computing*, 15:1119 – 1129, 1986.

[Kre71] G. Kreweras. Représentation polyédrique des préordres complets finis. In M. Barbut, editor, *Ordres Totaux Finis*. Paris, 1971.

[Law64] E. L. Lawler. On scheduling problems with deferral costs. *Management Science*, 11:280 – 288, 1964.

[Law77] E. L. Lawler. A 'pseudopolynomial' algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, 1:331 – 342, 1977.

[Law78] E. L. Lawler. Sequencing jobs to minimize total weighted completion time subject to precedence constraints. *Annals of Discrete Mathematics*, 2:75 – 90, 1978.

[LLRKS93] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. Sequencing and scheduling: Algorithms and complexity. In S. C. Graves, A. H. G. Rinnooy Kan, and P. H. Zipkin, editors, *Logistics of Production and Inventory*, volume 4 of *Handbooks in Operations Research and Management Science*, pages 445 – 522. North–Holland, Amsterdam, The Netherlands, 1993.

[Lov83] L. Lovász. Submodular functions and convexity. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming: The State of the Art – Bonn 1982*, pages 235 – 257. Springer, 1983.

[LQ92] J.–B. Lasserre and M. Queyranne. Generic scheduling polyhedra and a new mixed–integer formulation for single–machine scheduling. In E. Balas, G. Cornuéjols, and R. Kannan, editors, *Integer Programming and Combinatorial Optimization*, pages 136 – 149. Carnegie Mellon University, 1992. Proceedings of the 2nd International IPCO Conference.

[LRK78] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26:22 – 35, 1978.

[LRKB77] J. K. Lenstra, A. H. G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:343 – 362, 1977.

[MK92] M. Maes and B. Kappen. On the permutahedron and the quadratic placement problem. *Philips Journal of Research*, 46:267 – 292, 1992.

[MQW96] F. Margot, M. Queyranne, and Y. Wang. A branch-and-cut algorithm for single–machine precedence–constrained scheduling. Management Science Seminar, Faculty of Commerce, University of British Columbia, Vancouver, Canada, 1996.

[MR89] R. H. Möhring and F. J. Radermacher. The order–theoretic approach to scheduling : The deterministic case. In R. Slowinski and J. Weglarz, editors, *Advances in Project Scheduling*, pages 29 – 66. Elseviers Science Publishers, Amsterdam, 1989.

[MS93] F. Maffioli and A. Sciomachen. A mixed–integer model for solving ordering problems with side constraints. ECCO VI, Bruxelles, 1993.

[NS92] G. L. Nemhauser and M. W. P. Savelsbergh. A cutting plane algorithm for the single machine scheduling problem with release times. In M. Akgül, H. Hamacher, and S. Tufecki, editors, *Combinatorial Optimization: New Frontiers in the Theory and Practice*, volume 82 of *NATO ASI Series F: Computer and System Sciences*, pages 63 – 84. Springer, Berlin, 1992.

[NW88] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, New York, 1988.

[Pet88] R. Peters. L'ordonnancement sur une machine avec des contraintes de délai. *Belgian Journal of Operations Research, Statistics and Computer Science*, 28:33 – 76, 1988.

[Pot80] C. N. Potts. An algorithm for the single machine sequencing problem with precedence constraints. *Mathematical Programming Study*, 13:78 – 87, 1980.

[PQ78] J.–C. Picard and M. Queyranne. The time–dependent traveling salesman problem and its application to the tardiness problem in one–machine scheduling. *Operations Research*, 26:86 – 110, 1978.

[Pul89] W. R. Pulleyblank. Polyhedral combinatorics. In G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, editors, *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, pages 371 – 446. North–Holland, Amsterdam, 1989.

[QS95] M. Queyranne and A. S. Schulz. Scheduling unit jobs with compatible release dates on parallel machines with nonstationary speeds. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, number 920 in Lecture Notes in Computer Science, pages 307 – 320. Springer, Berlin, 1995. Proceedings of the 4th International IPCO Conference.

[Que93] M. Queyranne. Structure of a simple scheduling polyhedron. *Mathematical Programming*, 58:263 – 285, 1993.

[QW91a] M. Queyranne and Y. Wang. A cutting plane procedure for precedence–constrained single machine scheduling. Working paper. Faculty of Commerce, University of British Columbia, Vancouver, Canada, 1991.

[QW91b] M. Queyranne and Y. Wang. Single–machine scheduling polyhedra with precedence constraints. *Mathematics of Operations Research*, 16:1 – 20, 1991.

[QW92] M. Queyranne and Y. Wang. On the convex hull of feasible solutions to certain combinatorial problems. *Operations Research Letters*, 11:1 – 11, 1992.

[QW93] M. Queyranne and Y. Wang. Hamiltonian path and symmetric travelling salesman polytopes. *Mathematical Programming*, 58:89 – 110, 1993.

[Rad52] R. Rado. An inequality. *The Journal of the London Mathematical Society*, 27:1 – 6, 1952.

[Sav85] M. W. P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4:285 – 305, 1985.

[Sch11] P. H. Schoute. *Analytical Treatment of the Polytopes Regularly Derived from the Regular Polytopes*. Johannes Müller, Amsterdam, 1911. Verhandelingen der Koninklijke Akademie van Wetenschappen XI.3.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Chichester, 1986.

[Sch93a] A. S. Schulz. Polyedrische Charakterisierung von Scheduling Problemen. Diploma Thesis, Department of Mathematics, Technical University of Berlin, Berlin, Germany, 1993.

[Sch93b] A. S. Schulz. Single machine scheduling subject to N–sparse precedence constraints (extended abstract). In U. Faigle and C. Hoede, editors, *3rd Twente Workshop on Graphs and Combinatorial Optimization*, pages 166 – 169. Faculty of Applied Mathematics, University of Twente, The Netherlands, June 1993. Memorandum No. 1132.

[Sch95] A. S. Schulz. The permutahedron of series–parallel posets. *Discrete Applied Mathematics*, 57:85 – 90, 1995.

[Sch96a] A. S. Schulz. *Polytopes and Scheduling*. PhD thesis, University of Technology, Berlin, Germany, 1996.

[Sch96b] A. S. Schulz. Scheduling to minimize total weighted completion time: Performance guarantees of LP–based heuristics and lower bounds. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, number 1084 in Lecture Notes in Computer Science, pages 301 – 315. Springer, Berlin, 1996. Proceedings of the 5th International IPCO Conference.

[Sha71] L. S. Shapley. Cores of convex games. *International Journal of Game Theory*, 1:11 – 26, 1971.

[Sha93] J. F. Shapiro. Mathematical programming models and methods for production planning and scheduling. In S. C. Graves, A. H. G. Rinnooy Kan, and P. H. Zipkin, editors, *Logistics of Production and Inventory*, volume 4 of *Handbooks in Operations Research and Management Science*, pages 371 – 443. North–Holland, Amsterdam, 1993.

[Smi56] W. E. Smith. Various optimizers for single–stage production. *Naval Research and Logistics Quarterly*, 3:59 – 66, 1956.

[Sou89] J. P. Sousa. *Time Indexed Formulations of Non–Preemptive Single–Machine Scheduling Problems*. PhD thesis, Université Catholique de Louvain, Belgium, 1989.

[SS] A. S. Schulz and M. Skutella. Randomization strikes in scheduling: Improved approximations for minimizing the average weighted completion time. In preparation.

[SSW96] R. Schrader, A. S. Schulz, and G. Wambach. Base polytopes of series–parallel posets: Linear description and optimization. Report, Institute of Computer Science, University of Cologne, Cologne, Germany, 1996.

[SW92] J. P. Sousa and L. A. Wolsey. A time–indexed formulation of non–preemptive single–machine scheduling problems. *Mathematical Programming*, 54:353 – 367, 1992.

[SY92] J. G. Shanthikumar and D. D. Yao. Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Operations Research*, 40:293 – 299, 1992.

[Tsi92] J. N. Tsitsiklis. Special cases of traveling salesman and repairman problems with time windows. *Networks*, 22:263 – 282, 1992.

[Vel95] S. L. van de Velde. Dual decomposition of a single–machine scheduling problem. *Mathematical Programming*, 69:413 – 428, 1995.

[Wag59] H. M. Wagner. An integer programming model for machine scheduling. *Naval Research Logistics Quarterly*, 6:131 – 140, 1959.

[Wan96] Y. Wang. On the 3–approximation of scheduling with release dates. Technical report, Max–Planck–Institut für Informatik, Saarbrücken, Germany, June 1996.

[Wol90a] L. A. Wolsey. Formulating single machine scheduling problems with precedence constraints. In J. J. Gabszewicz, J.–F. Richard, and L. A. Wolsey, editors, *Economic Decision–Making: Games, Econometrics and Optimisation*, pages 473 – 484. North–Holland, Amsterdam, 1990.

[Wol90b] L. A. Wolsey. Valid inequalities for 0–1 knapsacks and mips with generalised upper bound constraints. *Discrete Applied Mathematics*, 29:251 – 261, 1990.

[YKK84] V. A. Yemelichev, M. M. Kovalev, and M. K. Kravtsov. *Polytopes, Graphs and Optimisation*. Cambridge University Press, 1984.

[Zie95] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer, New York, 1995.