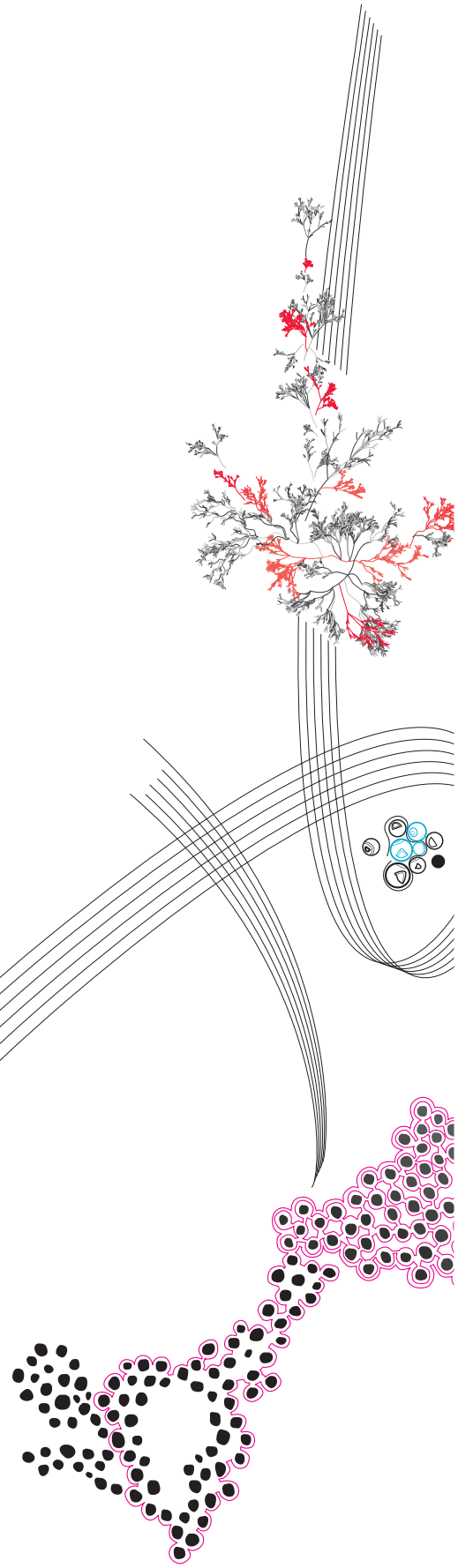

BSc Thesis Applied Mathematics

Scheduling-Polyhedron with a Time-Horizon

Sytze Arend Jacobus Horjus

Supervisor: M.Walter

July, 2024

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Scheduling-Polyhedron with a Time-Horizon

S.A.J. Horjus*

July, 2024

Abstract

In a single machine scheduling problem where the machine can handle only one job at a time and is non preemptive, we look at schedules on a finite horizon $[0, T]$. These schedules are defined by a vector containing the completion times of each job. For positive processing times, a polyhedron P can be constituted that completely describes the convex hull of these feasible schedules. In this thesis we find this P and prove that it is the convex hull of all feasible schedules on the horizon $[0, T]$

Keywords: linear program, job scheduling, polyhedra, single-machine scheduling

1 Introduction

Imagine we have a single machine processing a set J of n jobs. We are given the processing time vector p where each element $p_j > 0$ denotes the time the machine needs to process job $j \in J$. This means that when the processing of any job $j \in J$ starts, exactly p_j units of time later job j is completed. The machine can process only one job at a time and can not be interrupted by a job once it is processing another job. Before execution of the machine the order of the jobs is determined, we store this order in a vector that is called a schedule, where every component denotes the completion time of job j. We will define this schedule more formally later.

Which schedule to choose for an execution of a single machine is an interesting problem. In the literature this problem is called a single machine scheduling problem, which is a special case of the class of optimal job scheduling problems. A lot of research has been done for optimal job scheduling. Actually so much, that a whole notation has been developed to meaningfully name all different kind of job scheduling problems [2].

Smith, for example, showed in 1956 in [5] how to solve a $1||\sum w_j C_j$ problem, i.e., minimising the weighted sum of completion times. But there are also other interesting directions on this single machine scheduling problem. Queyranne [3] found a linear programming (LP) formulation that bounds all possible schedules on an infinite horizon, i.e., $[0, \infty)$. And together with Wang [4], he elaborated later on this problem by also adding the option of having precedences of jobs.

Why is research on single machine scheduling jobs important? Firstly, it is easy to see that optimal job scheduling problems have a wide variety of usage. From the production of products to the scheduling of tasks on a computer, these problems are relevant. When researching single machine scheduling problems one has the option to find underlying structure which can be extended to more complicated problems. For example, Smith's

*Email: S.A.J.horjus@student.utwente.nl

paper on finding the minimum weighted sum works on a single machine, but for multiple machines it won't work. Finding bounds for the possible schedules works for both single and multiple machines. For single machines this is exactly what Queyranne investigated [3].

But Queyranne investigated the case for schedules on the infinite horizon, while in practice we are more interested in bounds for schedules on the finite horizon $[0, T]$.

In this thesis we will close this gap by showing that on a finite horizon $[0, T]$, certain inequalities make a complete linear programming (LP) formulation for this problem. We will do this by showing that the intersection of the LP formulations for the horizons $[0, \infty)$ and $(-\infty, T]$ constitutes the complete LP formulation using proof techniques from [1].

2 Definitions and preliminary results

To be able to formulate our problem more precisely, we have to define a few things, which we will do in this section. We will also prove some lemmas and a proposition in this section to make the proof in the next section shorter and easier to understand.

We start by stating how we keep track of when each job is being processed. In the literature starting times are often used for modeling purposes and completion times when intuitive notation is preferred. However, in this thesis we will make use of half times. By the half time of a job we mean the time at which half of a job has been processed by the machine, i.e., the average between the starting and completion time of a job. The advantage of using half times will become clear later in this section.

We now use these half times in the definition of a feasible schedule on the finite horizon $[0, T]$.

Definition 2.1. *A vector $h \in \mathbb{R}^n$, where each element h_j denotes the half time of job j, is called a **feasible schedule** of all jobs on a finite time horizon $[0, T]$ if it satisfies the following inequalities*

$$h_j \geq \frac{p_j}{2}, \text{ for all } j \in J$$
$$h_j \leq T - \frac{p_j}{2}, \text{ for all } j \in J$$
$$h_j \geq h_k + \frac{p_k}{2} + \frac{p_j}{2} \quad \text{or} \quad h_k \geq h_j + \frac{p_j}{2} + \frac{p_k}{2}, \quad \forall j, k \in J, j \neq k.$$

$H^{0,T}$ denotes the set of all feasible schedules on the finite time horizon $[0, T]$.

Note that these inequalities in the definition of a feasible schedule imply that $H^{0,T} = \varnothing$ if $T < \sum_{j \in J} p_j$.

The following Proposition is a generalised result from Smith's Rule [5], which in addition to the classical Smith's Rule, also allows for negative weights. This result will be essential this thesis.

Proposition 2.2 (Smith's Rule)**.** *Let $w \in \mathbb{R}^n$ and $T \geq \sum_{j \in J} p_j$. Then a feasible schedule $h \in H^{0,T}$ minimizes $\sum_{j \in J} w_j h_j$ if and only if*

$$\frac{w_i}{p_i} \geq \frac{w_j}{p_j} \text{ holds for all } i, j \in J \text{ with } h_i < h_j \tag{2.1}$$

and

$$\text{all idle time is in between job } k \text{ and } k+1 \text{ in } J \text{ for which } \frac{w_k}{p_k} \geq 0 \text{ and } \frac{w_{k+1}}{p_{k+1}} < 0. \tag{2.2}$$

Proof. We can look at the idle times as jobs with a positive processing time but with corresponding weight being zero, since they do not add anything to the objective value. Then, let the total idle time $T-\sum_{j\in J} p_j$ be distributed over $k \leq n-1$ jobs (since there is at max room for $n-1$ spots of idle times in a schedule) and let these k 'idle' jobs be contained in I. Then the total job set $J' = J \cup I$ contains $n+k$ jobs, such that $\sum_{j\in J'} p_j = T$, meaning all jobs in J' are scheduled directly after each other.
Firstly, we prove that for all $k \leq n-1$ and all distributions of idle time among these k jobs, that $g \in \mathbb{R}^{n+k}$ is an optimal feasible schedule if and only if

$$\frac{w_i}{p_i} \geq \frac{w_j}{p_j} \text{ holds for all } i,j \in J' \text{ with } g_i < g_j. \tag{2.3}$$

($\Longrightarrow$) Suppose now we have a feasible schedule $g \in \mathbb{R}^{n+k}$ containing the half times of all jobs in J', that does not satisfy (2.3) but is optimal. Then there must exist jobs $i,j \in J'$ such that $\frac{w_i}{p_i} < \frac{w_j}{p_j}$ while $g_i < g_j$ and are scheduled directly after each other. Now suppose a feasible schedule g' that is equal to g except that the order of jobs i and j is flipped, thus $g'_i > g'_j$. Then the objective value change is

$$\begin{aligned}&\sum_{j\in J'} w_j g'_j - \sum_{j\in J'} w_j g_j = w_i(g'_i - g_i) + w_j(g'_j - g_j)\\ =&w_i(p_j + \frac{p_i}{2} - \frac{p_i}{2}) + w_j(\frac{p_j}{2} - p_i - \frac{p_j}{2}) = w_i p_j - w_j p_i < 0,\end{aligned}$$

Where that last inequality is true since $\frac{w_i}{p_i} < \frac{w_j}{p_j}$ implies $w_i p_j < w_j p_i$. This contradicts optimality of g.

($\Longleftarrow$) For the reverse direction, following the same logic, we see that if we have a feasible schedule $g \in \mathbb{R}^{n+k}$ satisfying (2.3), then swapping any two jobs $i,j \in J'$, that are scheduled next to each other, results in a higher objective value, except when $\frac{w_i}{p_i} = \frac{w_j}{p_j}$. Then swapping the two jobs does not change the objective value.

Lastly (2.3) implies (2.2), because the weights of the 'idle' jobs $j \in I$ are all zero. This concludes the proof.

■

The following results will play a big part in constructing the LP in the next section.

Definition 2.3. $g(S) \coloneqq \min\{\sum_{j\in S} p_j h_j : h \in H^{0,T}\}$ *for* $S \subseteq J$.

A useful result of this definition is presented in the following lemma.

Lemma 2.4. *For all* $S \subseteq J$ *we have*

$$g(S) = \min\{\sum_{j\in S} p_j h_j : h \in H^{0,T}\} = \frac{1}{2}(\sum_{j\in S} p_j)^2.$$

Proof. Let us try to find $h^* \in H^{0,T}$ that is optimal for the minimisation problem. We define weights such that

$$w_j \coloneqq \begin{cases} p_j & \text{for } j \in S \\ 0 & \text{for } j \notin S. \end{cases}$$

We then see that

$$\min\{\sum_{j\in S} p_j h_j : h \in H^{0,T}\} = \min\{\sum_{j\in J} w_j h_j : h \in H^{0,T}\}.$$

Since the weights are all non-negative, by Proposition 2.2 we know that $\sum_{j\in J} w_j h_j$ is minimised for a feasible schedule h^* if the only idle time is after all processed jobs and the jobs in S are ordered by $\frac{w_j}{p_j}$ from high to low. Note, $\frac{w_j}{p_j}$ is either 1 (if $j \in S$) or 0 (if $j \notin S$). This means putting the jobs in S first, the jobs in $J\backslash S$ second and the idle times last. Assuming $S = \{s_1, s_2, \ldots, s_k\} \subseteq J$, then any such schedule results in the following objective value

$$\begin{aligned}
&w_{s_1}\frac{1}{2}p_{s_1} + w_{s_2}(p_{s_1} + \frac{1}{2}p_{s_2}) + \cdots + w_{s_k}(p_{s_1} + \cdots + p_{s_{k-1}} + \frac{1}{2}p_{s_k}) + 0 \\
&= p_{s_1}\frac{1}{2}p_{s_1} + p_{s_2}(p_{s_1} + \frac{1}{2}p_{s_2}) + \cdots + p_{s_k}(p_{s_1} + \cdots + p_{s_{k-1}} + \frac{1}{2}p_{s_k}) \\
&= \frac{1}{2}(p_{s_1} + p_{s_2} + \cdots + p_{s_k})^2 = \frac{1}{2}(\sum_{j\in S} p_j)^2.
\end{aligned}$$

■

Should we have used completion times instead of half times, then $g(S) = \frac{1}{2}((\sum_{j\in S} p_j)^2 + \sum_{j\in S} p_j^2)$ [4], which is harder to work with than the outcome of Lemma 2.4. Similarly, starting times will also gives a result, which is harder to work with. This shows us already an advantage of using half times.

Lemma 2.5. *For all $S \subseteq J$,*

$$\max\{\sum_{j\in S} p_j h_j : h \in H^{0,T}\} = -g(S) + T\sum_{j\in S} p_j.$$

Proof. We know that

$$\max\{\sum_{j\in S} p_j h_j : h \in H^{0,T}\} = -\min\{\sum_{j\in S} -p_j h_j : h \in H^{0,T}\}$$

Let us try to find $h^* \in H^{0,T}$ that is optimal for the minimisation problem $\min\{\sum_{j\in S} -p_j h_j : h \in H^{0,T}\}$. We define weights such that

$$w_j \coloneqq \begin{cases} -p_j & \text{for } j \in S \\ 0 & \text{for } j \notin S. \end{cases}$$

We then see that

$$-\min\{\sum_{j\in S} -p_j h_j : h \in H^{0,T}\} = -\min\{\sum_{j\in J} w_j h_j : h \in H^{0,T}\}.$$

Since the weights are all negative, by Proposition 2.2 we know that $\sum_{j\in J} w_j h_j$ is minimised for a feasible schedule h^* if the only idle time is before all jobs and the jobs are ordered by $\frac{w_j}{p_j}$ from high to low. Note, $\frac{w_j}{p_j}$ is either -1 (if $j \in S$) or 0 (if $j \notin S$).

This means putting the idle times and jobs in $J \backslash S$ first and the jobs in S last. Assuming $S = \{s_1, s_2, \ldots, s_k\} \subseteq J$, then any such schedule results in the following objective value

$$\begin{aligned}
&- (0 + w_{s_1}(T - \frac{1}{2}p_{s_1} - p_{s_2} - \cdots - p_{s_k}) \\
&\qquad + w_{s_2}(T - \frac{1}{2}p_{s_2} - p_{s_3} - \cdots - p_{s_k}) + \cdots + w_{s_k}(T - \frac{1}{2}p_{s_k})) \\
&= -(0 - p_{s_1}(T - \frac{1}{2}p_{s_1} - p_{s_2} - \cdots - p_{s_k}) \\
&\qquad - p_{s_2}(T - \frac{1}{2}p_{s_2} - p_{s_3} - \cdots - p_{s_k}) - \cdots - p_{s_k}(T - \frac{1}{2}p_{s_k})) \\
&= 0 + p_{s_1}(T - \frac{1}{2}p_{s_1} - p_{s_2} - \cdots - p_{s_k}) \\
&\qquad + p_{s_2}(T - \frac{1}{2}p_{s_2} - p_{s_3} - \cdots - p_{s_k}) + \cdots + p_{s_k}(T - \frac{1}{2}p_{s_k}) \\
&= T\sum_{j \in S} p_j - (p_{s_1}(\frac{1}{2}p_{s_1} + p_{s_2} + \cdots + p_{s_k}) \\
&\qquad + p_{s_2}(\frac{1}{2}p_{s_2} + p_{s_3} + \cdots + p_{s_k}) + \cdots + p_{s_k}(\frac{1}{2}p_{s_k})) \\
&= T\sum_{j \in S} p_j - \frac{1}{2}(\sum_{j \in S} p_j)^2 = T\sum_{j \in S} p_j - g(S).
\end{aligned}$$

Thus we have shown that

$$\max\{\sum_{j \in S} p_j h_j : h \in H^{0,T}\} = -g(S) + T\sum_{j \in S} p_j.$$

■

Also in this proof we can see the advantage of using half times over completion times or starting times. Describing half times from the endpoint T instead of the starting point 0 works out nicely and gives a nice short result. This works because of the symmetric nature of half times.

3 Main result

Inspired by Queyranne's research [3] on this topic, we define the following polyhedra

$$\begin{aligned} P^{0,\infty} &= \{h \in \mathbb{R}^n : \sum_{j\in S} p_j h_j \geq g(S) \quad \text{for all } \varnothing \neq S \subseteq J\} \\ P^{-\infty,T} &= \{h \in \mathbb{R}^n : \sum_{j\in S} p_j h_j \leq -g(S) + T\sum_{j\in S} p_j \quad \text{for all } \varnothing \neq S \subseteq J\} \\ P &= P^{0,\infty} \cap P^{-\infty,T}. \end{aligned} \tag{3.1}$$

The main theorem of this paper is the following theorem.

Theorem 3.1 (Main Theorem)**.** *Let J be a set of n jobs, let $p_j > 0$ be the processing time of job $j \in J$ and $T \geq \sum_{j\in J} p_j$, then*

$$conv(H^{0,T}) = P,$$

where P is the polyhedron defined in (3.1)*.*

In order to prove Theorem 3.1 we consider a bijective linear mapping Π to transform our P and $H^{0,T}$ into something that is easier to work with. We try $\Pi(x) = p \odot x$, where $\odot$ is the element-wise product and p is the processing time vector. Then given $p \in \mathbb{R}^n$, the mapping Π gives us

$$X \coloneqq \Pi(H^{0,T}) = \{p \odot h : h \in H^{0,t}\} \tag{3.2}$$

and

$$\begin{aligned} Q &\coloneqq \Pi(P) \\ &= \{\Pi(h) | h \in P\} \\ &= \{p \odot h | h \in \mathbb{R}^n, \sum_{j\in S} p_j h_j \geq g(S) \text{ and } \sum_{j\in S} p_j h_j \leq -g(S) + T\sum_{j\in S} p_j \quad \text{for all } \varnothing \neq S \subseteq J\} \\ &= \{p \odot h | h \in \mathbb{R}^n, \sum_{j\in S} (p \odot h)_j \geq g(S), \sum_{j\in S} (p \odot h)_j \leq -g(S) + T\sum_{j\in S} p_j \quad \text{for all } \varnothing \neq S \subseteq J\}. \end{aligned}$$

Now with the substitution of $x = p \odot h$ we get

$$Q = \{x \in \mathbb{R}^n | \sum_{j\in S} x_j \geq g(S) \text{ and } \sum_{j\in S} x_j \leq -g(S) + T\sum_{j\in S} p_j\}. \tag{3.3}$$

Indeed we see that it easier to use these mapped sets, since Theorem 4.29 in [1] uses almost the exact same polyhedron as Q. Therefore, to prove Theorem 3.1 we first prove the following theorem using similar proof techniques as in [1].

Theorem 3.2. *Let J be a set of n jobs, let $p_j > 0$ be the processing time of job $j \in J$ and $T \geq \sum_{j\in J} p_j$, then*

$$Q = conv(X),$$

where X and Q are defined in (3.2) *and* (3.3)*, respectively.*

Proof of Theorem 3.2. To prove that $Q = conv(X)$ it is enough to show that for all $c \in \mathbb{R}^n$

$$\min_{x \in X} c^\intercal x \overset{!}{=} \min_{x \in Q} c^\intercal x \tag{3.4}$$

holds. Without loss of generality, reorder the jobs such that $c_j \geq c_{j+1}$ for all $j \in J \backslash \{n\}$. Let $\overline{x} = p \odot \overline{h}$, where $\overline{h}_j = \begin{cases} \frac{p_j}{2} + \sum_{i<j} p_i & \text{if } c_j \geq 0 \\ T - \frac{p_j}{2} - \sum_{i>j} p_i & \text{if } c_j < 0. \end{cases}$
Note that because of the definition of $\overline{h}$ and $T \geq \sum_{j=1}^n p_j$ that $\overline{h} \in H^{0,T}$, which implies that $\overline{x} \in X$. We claim that $\min_{x \in X} c^\intercal x = c^\intercal \overline{x} = \min_{x \in Q} c^\intercal x$.

To show that $c^\intercal \bar{x} = \min_{x \in Q} c^\intercal x$ or similarly that $\overline{x}$ is optimal for

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \sum_{j \in S} c_j x_j \\ \text{subject to} \quad & \sum_{j \in S} x_j \geq g(S) \quad \text{for all } S \subseteq J, \\ & \sum_{j \in S} x_j \leq -g(S) + T \sum_{j \in S} p_j \quad \text{for all } S \subseteq J, \end{aligned} \tag{3.5}$$

we first have to show that $\overline{x}$ is a feasible solution for this linear program.

From the definition of $g(S)$ it is clear that $\sum_{j \in S} \overline{x}_j \geq g(S)$ for all $S \subseteq J$. From Lemma 2.5 it is also clear that $\sum_{j \in S} \overline{x}_j \leq -g(S) + T \sum_{j \in S} p_j$ for all $S \subseteq J$. Thus $\overline{x}$ is feasible.

To show that $\overline{x}$ is optimal for the the linear program in (3.5) we use its dual, which is

$$\begin{aligned} \underset{y, y'}{\text{maximize}} \quad & \sum_{S \subseteq J} g(S) y_S + (-g(S) + T \sum_{j \in S} p_j) y'_S \\ \text{subject to} \quad & \sum_{S \ni j} y_S + y'_S = c_j \quad \text{for all } j \in J, \\ & y_S \geq 0 \quad \text{for all } S \subseteq J, \\ & y'_S \leq 0 \quad \text{for all } S \subseteq J. \end{aligned}$$

Now we define two special subsets of J as follows

$$S_j \coloneqq \begin{cases} \varnothing, & \text{if } j = 0 \\ \{1, \ldots, j\}, & \text{if } j = 1, 2, \ldots, n \end{cases}$$

$$S'_j \coloneqq \begin{cases} \varnothing, & \text{if } j = n+1 \\ \{j, \ldots, n\}, & \text{if } j = 1, 2, \ldots, n. \end{cases}$$

Let $k \in \{0, 1, \ldots, n\}$ be the unique number such that $c_k \geq 0$ and $c_{k+1} < 0$. If all elements of c are positive then $k = n$ and if they are all negative then $k = 0$. We formulate a dual solution $(\overline{y}, \overline{y}')$ as follows

$$\overline{y}_S \coloneqq \begin{cases} c_j - c_{j+1}, & \text{for } S = S_j, \quad j = 1, \ldots, k-1 \\ c_k, & \text{for } S = S_k \\ 0, & \text{otherwise} \end{cases}$$

$$\overline{y}'_S \coloneqq \begin{cases} c_j - c_{j-1}, & \text{for } S = S'_j, \quad j = k+2, \ldots, n \\ c_{k+1}, & \text{for } S = S'_{k+1} \\ 0, & \text{otherwise.} \end{cases}$$

It is easy to see that $\overline{y}'_S \leq 0$ and $\overline{y}_S \geq 0$. In addition, since the following holds

$$\begin{aligned} \sum_{S \ni j} y_S + y'_S &= \begin{cases} \sum_{i=j}^{k} y_{S_j} & \text{for } j \leq k \\ \sum_{i=k+1}^{j} y'_{S'_j} & \text{for } j \geq k+1 \end{cases} \\ &= \begin{cases} (c_j - c_{j+1}) + \cdots + (c_{k-1} - c_k) + c_k & \text{for } j \leq k \\ c_{k+1} + (c_{k+2} - c_{k+1}) + \cdots + (c_j - c_{j-1}) & \text{for } j \geq k+1 \end{cases} \\ &= c_j, \end{aligned}$$

$(\overline{y}, \overline{y}')$ turns out to be a feasible solution to the dual.

Now we can start showing equality of the two objective functions using the defined dual and primal solutions. We start with splitting up the primal objective function as follows

$$\sum_{j=1}^{n} c_j \overline{x}_j = \sum_{j=1}^{k} c_j (p_j (\frac{p_j}{2} + \sum_{i<j} p_i)) + \sum_{j=k+1}^{n} c_j (p_j (T - \frac{p_j}{2} - \sum_{i>j} p_i)). \tag{3.6}$$

The first part of equation (3.6) becomes

$$\begin{aligned} &\sum_{j=1}^{k} c_j (p_j (\frac{p_j}{2} + \sum_{i<j} p_i)) \\ =&\sum_{j=1}^{k} c_j (\frac{1}{2}(2p_1 p_j + 2p_2 p_j + \cdots + 2p_{j-1} p_j + p_j^2)) \\ =&\sum_{j=1}^{k} c_j (\frac{1}{2}(p_1 + p_2 + \cdots + p_j)^2 - \frac{1}{2}(p_1 + p_2 + \cdots + p_{j-1})^2) \\ =&\sum_{j=1}^{k} c_j (g(S_j) - g(S_{j-1})) \\ =&g(S_k) c_k + \sum_{j=1}^{k-1} g(S_j)(c_j - c_{j+1}) \\ =&\sum_{j=1}^{k} g(S_j) \overline{y}_S = \sum_{S \subseteq J} g(S) \overline{y}_S. \end{aligned}$$

The second part of equation (3.6) becomes

$$\begin{aligned}
&\sum_{j=k+1}^{n} c_j(p_j(T - \frac{p_j}{2} - \sum_{i>j} p_i)) \\
&= \sum_{j=k+1}^{n} c_j(p_jT + p_j(-\frac{p_j}{2} - p_{j+1} - \cdots - p_n)) \\
&= \sum_{j=k+1}^{n} c_j(p_jT - \frac{1}{2}(p_j^2 + 2p_{j+1}p_j + \cdots + 2p_np_j)) \\
&= \sum_{j=k+1}^{n} c_j(p_jT - \frac{1}{2}((p_j + p_{j+1} + \cdots + p_n)^2 - (p_{j+1} + p_{j+2} + \cdots + p_n)^2)) \\
&= \sum_{j=k+1}^{n} c_j(p_jT + \frac{1}{2}(p_{j+1} + p_{j+2} + \cdots + p_n)^2 - \frac{1}{2}(p_j + p_{j+1} + \cdots + p_n)^2) \\
&= \sum_{j=k+1}^{n} c_j(p_jT + g(S'_{j+1}) - g(S'_j)) \\
&= \sum_{j=k+1}^{n} c_jp_jT + \sum_{j=k+1}^{n} c_j(g(S'_{j+1}) - g(S'_j)) \\
&= \sum_{j=k+1}^{n} c_jp_jT - c_{k+1}g(S_{k+1}) - \sum_{j=k+2}^{n} g(S'_j)(c_j - c_{j-1}) \\
&= \sum_{j=k+1}^{n} c_jp_jT + \sum_{j=k+1}^{n} -g(S'_j)\overline{y}'_S.
\end{aligned}$$

Since $(\overline{y}, \overline{y}')$ is a dual solution we know that $\sum_{S \ni j} \overline{y}_S + \overline{y}'_S = c_j$, therefore

$$\begin{aligned}
\sum_{j=k+1}^{n} c_jp_jT + \sum_{j=k+1}^{n} -g(S'_j)\overline{y}'_S &= \sum_{j=k+1}^{n} (p_jT \sum_{S \ni j} \overline{y}'_S) + \sum_{j=k+1}^{n} -g(S'_j)\overline{y}'_S \\
&= T \sum_{S \subseteq J} \sum_{j \in S} p_j\overline{y}'_S + \sum_{S \subseteq J} -g(S)\overline{y}'_S \\
&= \sum_{S \subseteq J} (-g(S) + T \sum_{j \in S} p_j)\overline{y}'_S.
\end{aligned}$$

Conclusively

$$\begin{aligned}
\sum_{j=1}^{n} c_j\overline{x}_j &= \sum_{j=1}^{k} c_j(p_j(\frac{p_j}{2} + \sum_{i<j} p_i)) + \sum_{j=k+1}^{n} c_j(p_j(T - \frac{p_j}{2} - \sum_{i>j} p_i)) \\
&= \sum_{S \subseteq J} g(S)\overline{y}_S + (-g(S) + T \sum_{j \in S} p_j)\overline{y}'_S.
\end{aligned}$$

But this is exactly the objective function of the dual. Since the two objective values from the primal and dual have the same value, $\overline{x}$ must be optimal for the primal in (3.5).

Now to show that $\overline{x}$ is optimal for the left hand side of equation (3.4), i.e. show that $c^\intercal \overline{x} = \min_{x \in X} c^\intercal x$, we let $w_j = c_j p_j$. Then by Proposition 2.2 the schedule that minimizes $\sum_{j \in J} w_j h_j$ is ordered such that $c_1 \geq c_2 \geq ... \geq c_n$ and has only one idle time between the job k and $k+1$ for which $c_k \geq 0$ and $c_{k-1} < 0$. The schedule that does exactly this is $\overline{h}$. Now since $X = \{p \odot h : h \in H^{0,T}\}$ and $c_j = \frac{w_j}{p_j}$, $\min_{x \in X} c^T x = \min_{h \in H^{0,T}} w^\intercal h$. Therefore $\overline{h}$ minimizes $w^\intercal h$ implies that $\overline{x} = p \odot \overline{h}$ minimizes $c^\intercal x$ over X.

This means that we have shown that

$$\min_{x \in X} c^\intercal x = c^\intercal \overline{x} = \min_{x \in Q} c^\intercal x,$$

which implies that $Q = conv(X)$. ■

Now that we have shown that $Q = conv(X)$ it remains to show that $P = conv(H^{0,T})$.

Proof of Theorem 3.1. By Theorem 3.2 we know that $Q = conv(X)$. Then $\Pi^{-1}(Q) = \Pi^{-1}(conv(X))$ must also be true. Since Π is a linear bijective mapping and $X = \Pi(H^{0,T})$ by equation (3.2), we see that $\Pi^{-1}(conv(X)) = conv(\Pi^{-1}(X)) = conv(H^{0,T})$. Similarly, since Π is a linear bijective mapping and $Q = \Pi(P)$ by equation (3.3) , $\Pi^{-1}(Q) = P$. Consequently $P = conv(H^{0,T})$. ■

4 Conclusion

In this thesis we demonstrated that the polyhedron in (3.1) is a complete LP formulation of the convex hull of all feasible schedules on the finite time horizon $[0, T]$. We chose to use half times and they came in very helpful for proving. This proof could have also used completion times or starting times, but we showed it was easiest this way.

Considering that all schedules on the finite horizon are possible, this result remains still very general. For future research one might look at schedules with special constraints and use methods in the thesis to find its corresponding polyhedron and prove that it is complete. For example, in [4] a polyhedron was found for schedules with precedence constraints on the infinite horizon. One could try to see if it is easy to apply techniques used in this thesis on this problem for a finite horizon $[0, T]$. Additionally, it might be worth investigating other single-machine scheduling, such as those involving release times or deadlines, using the methods in this thesis

Furthermore, our findings have implications beyond single-machine scheduling. They could contribute to establishing optimal bounds for schedules involving multiple machines. Lastly, it would also be interesting what would happen if one makes the T a variable such that the horizon $[0, T]$ fluctuates. Investigating how the polyhedron evolves under such conditions could yield valuable insights.

References

[1] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. *Integer Programming*, volume 271 of *Graduate Texts in Mathematics*. Springer International Publishing.

[2] 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. In P. L. Hammer, E. L. Johnson, and B. H. Korte, editors, *Annals of Discrete Mathematics*, volume 5 of *Discrete Optimization II*, pages 287–326. Elsevier.

[3] Maurice Queyranne. Structure of a simple scheduling polyhedron. 58(1):263–285.

[4] Maurice Queyranne and Yaoguang Wang. Single-machine scheduling polyhedra with precedence constraints. 16(1):1–20. Publisher: INFORMS.

[5] Wayne E. Smith. Various optimizers for single-stage production. 3(1):59–66. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800030106.