20th International Congress on Modelling and Simulation, Adelaide, Australia, 1–6 December 2013
www.mssanz.org.au/modsim2013

A Variable Sized Bucket Indexed Formulation for Nonpreemptive Single Machine Scheduling Problems

R. Clement [a], **N. Boland** [a], **H. Waterer** [a]

[a] *Centre for Optimal Planning and Operations, University of Newcastle, Australia*
Email: riley.clement@uon.edu.au

Abstract: The single machine scheduling problem (SMSP) is a classic problem in optimisation which has been extensively studied over the past 60 years. To solve an instance of the problem a set of jobs must be scheduled on a single machine, so that at any time the machine is either idle or processing exactly one job. We consider a nonpreemptive version of this problem which requires that the processing of a job continue uninterrupted for the duration of its processing time. Several mixed integer linear programs exist for the SMSP, with the classic time indexed (TI) model being the most common formulation. The TI formulation can be applied to a range of SMSP variations, with all standard min-sum scheduling criteria capable of being expressed as linear functions of the TI variables. Many complex production planning problems are modelled using TI variables, and so the TI formulation often serves as natural basis for designing mixed integer linear programs to solve these problems. To formulate the TI model the problem data is assumed to be integer and a sufficiently large planning horizon is discretised into time periods of unit length. The length of the planning horizon can be no smaller than the sum of all processing times, and hence grows pseudopolynomially with the size of the problem input. The TI formulation is known to have a strong linear relaxation compared to alternative formulations, however for instances where the sum of processing times is large the resulting model may be intractable due to the large number of constraints and variables.

The authors recently proposed a mixed integer linear program, named the bucket indexed (BI) formulation, for which the time horizon is discretised into periods of the same length and no larger than the processing time of the shortest job. The BI model generalises the TI model to one in which either at most two or three jobs can be processing in each period. In this paper we present a model, named the variable sized bucket indexed (BI-VAR) formulation, in which the lengths of the periods are not required to be identical. This model generalises the BI model to one in which each period is characterised as either permitting at most two, three, or an arbitrary number of jobs to be processed within it. In addition we present necessary conditions for a partition of the time horizon to be valid for the BI-VAR model.

Keywords: *Single machine scheduling, mixed integer linear programming, time indexed formulations*

1 INTRODUCTION

The single machine scheduling problem (SMSP) is a classic problem in optimisation which has been extensively studied over the past 60 years. To solve an instance of the problem a set of jobs J must be scheduled on a single machine, so that at any time the machine is either idle or processing exactly one job. Each job $j \in J$ has a processing time p_j and may also have a release date r_j or a deadline $\bar{d}_j$ which restricts the interval of time in which the job may be scheduled. We consider a nonpreemptive version of this problem which requires that the processing of a job continue uninterrupted for the duration of its processing time. Several standard min-sum criteria which are often used to evaluate schedules also require each job $j \in J$ to have a due date d_j. For a comprehensive review of scheduling problems, we refer the reader to the survey paper by Lawler et al. (1993) or the texts Brucker (2007) and Pinedo (2002).

Several mixed integer linear programs exist for the SMSP, with the classic time indexed (TI) model, introduced by Sousa and Wolsey (1992) being the most common formulation. The TI formulation can be applied to a range of SMSP variations, with all standard min-sum scheduling criteria capable of being expressed as linear functions of the TI variables. Many complex production planning problems are modelled using TI variables, and so the TI formulation often serves as natural basis for designing mixed integer linear programs for solving these problems.

To formulate the TI model the problem data is assumed to be integer and a sufficiently large planning horizon is discretised into time periods of unit length. The length of the planning horizon can be no smaller than the sum of all processing times, and hence grows pseudopolynomially with the size of the problem input. The TI model is known to have a strong linear relaxation compared to alternative formulations (Dyer and Wolsey (1990)), however for instances where the sum of processing times is large, the resulting model may be intractable due to the large number of constraints and variables. Khowala et al. (2005) computationally verified that solution times obtained with the TI model grew exponentially with the sum of processing times and in some cases the LP relaxation may not even be solved in a reasonable time (Van den Akker et al. (1999), Baptiste and Sadykov (2009)).

Recently, two exact approaches have been proposed for formulating mixed integer linear programs with a coarser discretisation of the planning horizon. The first, proposed in Sadykov (2006) and Baptiste and Sadykov (2009) requires that the partition which gives rise to the coarser periods (or intervals) is restricted to be a superset of any release dates and due dates. By ensuring that the partition satisfies particular criteria the ordering of jobs within each interval can be determined by appealing to Smith's rule (Smith (1956)). The resulting model is called the Interval Indexed Formulation, which should not be confused with the Interval Indexed Formulation of Hall et al. (1997).

The second recent approach is a mixed integer linear program, proposed by the authors of this paper (Boland et al. (2013)), which we refer to as the bucket indexed (BI) formulation. The BI model is formulated across a time horizon which is partitioned into time periods of equal length called buckets. The use of the term *bucket* is motivated by ideas in lot sizing. Pochet and Wolsey (2006) (p374) define small bucket models of production planning and scheduling problems to be those where the machine setup status remains constant during each period/bucket. The discretisation used in the TI model is consistent with this definition. Big bucket problems are those in which buckets may accommodate the production of more than one type of item. In the BI model the length of the buckets is a parameter of the model and can be chosen large enough to allow at most, the processing of two jobs within each bucket. The BI model generalises the TI model and the two are equivalent if the problem data is integer and a bucket is of unit length. Under a coarser partition the BI model may have considerably less variables than the TI model and a much sparser constraint matrix.

In this study we propose a variable sized bucket indexed (BI-VAR) formulation, another mixed integer linear program which generalises the previous BI model, and hence the TI model also. Similarly to the BI model, the BI-VAR model is formulated across a time horizon which is partitioned into buckets. The buckets, however, are not required to be of equal size. Furthermore, the length of a bucket may be large enough to permit three jobs to be processed within it, and if certain conditions are satisfied a maximum number of jobs which can be processed in the bucket is not imposed.

2 PRELIMINARIES

As with the TI model we will assume that the problem data is integer. Note that provided the data is rational, it can always be scaled until it is integer. We make the remark however, as is the case for the BI model, that it is not necessary to scale the data for use with the BI-VAR model.

We begin by supposing that the time horizon, $[0, T]$ is divided into a set of buckets, $B = \{1, \ldots, |B|\}$, by a partition $\{t_b : b \in B_0\}$ where $B_0 = B \cup \{0\}$, $t_0 = 0$ and $t_{|B|} = T$. We then define $I_b = [t_{b-1}, t_b)$ for all $b \in B$ such that bucket b corresponds to the right half-open real interval I_b. For convenience we define $\Delta_b = t_b - t_{b-1}$ for each $b \in B$ so that the length of interval I_b, which we also refer to as the size of bucket b, is Δ_b. We say job j starts in bucket b if $s_j \in I_b$, where s_j is the start of job j. Similarly, job j completes in bucket b if $c_j \in I_b$, where c_j is the completion time of job j. For each job j we define B_j to be the set of buckets in which job j may start. Membership to this set may be restricted by release dates and deadlines. Most studies define the due date with respect to completion time, that is if a job completes after its due date then some penalty is incurred. Since the processing times are fixed and the scheduling is nonpreemptive the due dates and deadlines can be equivalently defined with respect to start times, which we found to lead to a neater formulation. For each $b \in B_j$ we define $B^c_{jb} \subseteq B$ to be those buckets in which job j can finish, provided it started in bucket b. Formally, for $j \in J$, $b \in B_j$,

$$B^c_{jb} = \{b' \in B : \exists s \in I_b, s + p_j \in I_{b'}\}. \tag{1}$$

For $b \in B$ we also define $J_b = \{j \in J : b \in B^c_{jb}\}$ as the set of jobs which can both start and finish in bucket b.

Now suppose that job j starts in bucket b, at time t, and completes in bucket b', at time $t + p_j$. Furthermore, define $u = (t_b - t)/\Delta_b$ such that u is the fraction of bucket b contained in the interval $[t, t_b]$. If job j starts as early as possible then $t = \inf\{s \in I_b : s + p_j \in I_{b'}\} = \max\{t_{b-1}, t_{b'-1} - p_j\}$ and $u = \min\{t_b - t_{b-1}, t_b - t_{b'-1} + p_j\}/\Delta_b$. Since the intervals corresponding to buckets are right half-open, it is not possible to consider the latest time at which job j can start in bucket b, however job j must start prior to $\sup\{s \in I_b : s + p_j \in I_{b'}\} = \min\{t_b, t_{b'} - p_j\}$. Under the assumption that all input data is integer, an optimal schedule will have integer start times for all jobs, and hence we can assume that if job j starts as late as possible (in an optimal schedule) then $t = \min\{t_b, t_{b'} - p_j\} - 1$ and $u = \max\{1, t_b - t_{b'} + p_j + 1\}/\Delta_b$. These upper and lowers limits for the value of u, given that job j starts in bucket b and completes in bucket b', are utilised in the formulation of the BI-VAR model and we denote them $u^u_{jbb'}$ and $u^l_{jbb'}$ respectively. Likewise we define $v^u_{jbb'}$ and $v^l_{jbb'}$ to be the upper and lower limits, respectively, for the value $v = (t' - t_{b'-1})/\Delta_{b'}$ if job j starts in bucket b and completes in bucket b' at time t'. Formally, for each $j \in J$, $b \in B_j$ and $b' \in B^c_{jb}$ we define:

$$u^l_{jbb'} = \max\{1, t_b - t_{b'} + p_j + 1\}/\Delta_b, \tag{2}$$
$$u^u_{jbb'} = \min\{t_b - t_{b-1}, t_b - t_{b'-1} + p_j\}/\Delta_b, \tag{3}$$
$$v^l_{jbb'} = \max\{0, t_{b-1} - t_{b'-1} + p_j\}/\Delta_{b'}, \tag{4}$$
$$v^u_{jbb'} = \min\{t_{b'} - t_{b'-1} - 1, t_b - t_{b'-1} + p_j - 1\}/\Delta_{b'}. \tag{5}$$

3 PARTITIONING THE TIME HORIZON

For the formulation of the BI-VAR model we require that each bucket $b \in B$ can be uniquely classified as belonging to either B^2, B^3 or B^∞, such that $B = [B^2|B^3|B^\infty]$. B^2 contains buckets in which no job can both start and finish and B^3 contains buckets in which no more than one job can start and finish. As a consequence no more than two jobs can be processed in buckets belonging to B^2 and no more than three jobs can be processed in buckets belonging to B^3. Each bucket $b \in B^\infty$ may permit any number of jobs provided that for each $j \in J_b$, the open interval $(t_{b-1}, t_b + p_j)$ does not contain the release date, due date or deadline of job j. The BI-VAR formulation includes capacity constraints for each of the three classes of buckets. Next we present the necessary conditions for a bucket to belong to B^2 or B^3, in Propositions 1 and 2 respectively.

3.1 At most two jobs per bucket (B^2)

Proposition 1. *Let bucket $b \in B$ be the interval $[t_{b-1}, t_b)$. If one of the following conditions is true for each $j \in J$ then there does not exist a feasible schedule in which any job both starts and finishes within b.*

$$t_b \leq r_j + p_j, \tag{6}$$
$$t_b \leq t_{b-1} + p_j. \tag{7}$$

Proof. Suppose that job j starts and finishes in bucket $b \in B$, at time s_j. If (6) is true then $s_j + p_j < r_j + p_j \Leftrightarrow s_j < r_j$ which cannot hold if the schedule respects release dates. Then $t_{b-1} \leq s_j < s_j + p_j < t_b$. If (7) is true then $s_j + p_j < t_{b-1} + p_j \Leftrightarrow s_j < t_{b-1}$, which is a contradiction. Hence if either (6) or (7) is true then job j cannot both start and finish in bucket b. □

3.2 At most three jobs per bucket (B^3)

Proposition 2. *Let bucket $b \in B$ be the interval $[t_{b-1}, t_b)$. If at least one of the following conditions is true for each ordered pair $(i, j) \in J \times J$, such that $i \neq j$ then there does not exist a feasible schedule in which two jobs both start and finish within b.*

$$t_b \leq r_j + p_j, \tag{8}$$
$$t_b \leq r_i + p_i + p_j, \tag{9}$$
$$t_b \leq t_{b-1} + p_i + p_j. \tag{10}$$

Proof. Suppose that jobs i and j both start and finish in bucket $b \in B$. Let the start times of i and j be s_i and s_j respectively and, without loss of generality, assume that i precedes j. Then $t_{b-1} \leq s_i < s_i + p_i \leq s_j < s_j + p_j < t_b$. If (8) is true then $s_j + p_j < t_b \leq r_j + p_j \Leftrightarrow s_j + < r_j$ which cannot hold if the schedule respects release dates. If (9) is true then $s_i + p_i + p_j < t_b \leq r_i + p_i + p_j \Leftrightarrow s_i + < r_i$ which again cannot hold if the schedule respects release dates. If (10) is true then $s_i + p_i + p_j < t_b \leq t_{b-1} + p_i + p_j \Leftrightarrow s_i < t_{b-1}$, which is a contradiction. Hence if any of the conditions (8)–(10) are true then no two jobs can both start and finish in bucket b. □

4 A VARIABLE SIZED BUCKET INDEXED FORMULATION

To formulate the mixed integer linear program we define variables for each $j \in J$, $b \in B_j$ and $b' \in B^c_{jb}$:

$z_{jbb'} = 1$ if job j starts in bucket b and finishes in bucket b' and 0 otherwise.

$u_{jbb'} \in [1/\Delta, 1]$ takes value such that if j starts in bucket b and completes in bucket b', and 0 otherwise.

$v_{jbb'} \in [0, 1 - 1/\Delta]$ takes value such that if j starts in bucket b and completes in bucket b', and 0 otherwise.

If job j starts in bucket b and finishes in bucket b' then its start time, s_j, is given by $t_b - u_{jbb'}$, and its completion time, c_j, is given by $t_{b'-1} + v_{jbb'}$. Hence for each job $j \in J$ the start time and completion time can be determined using the following equations:

$$s_j = \sum_{b \in B_j} \sum_{b' \in B^c_{jb}} (t_b z_{jbb'} - \Delta_b u_{jbb'}) \text{ and} \tag{11}$$
$$c_j = \sum_{b \in B_j} \sum_{b' \in B^c_{jb}} (t_{b'-1} z_{jbb'} + \Delta_{b'} v_{jbb'}). \tag{12}$$

Note that in any valid solution we must have $s_j + p_j = c_j$ and the values of the v variables can be completely determined from the values of the z and u variables. Similarly the values of the u variables could be deduced from the values of the z and v variables. This redundancy is apparent in the following equality constraints which feature in the model. These equations could be used to project out the redundant variables leaving a more compact model, however we choose not to do so in favour of clarity:

$$\Delta_b u_{jbb'} + \Delta_{b'} v_{jbb'} = (t_b - t_{b'-1} + p_j) z_{jbb'}, \qquad j \in J, b \in B_j, b' \in B^c_{jb} \tag{13}$$

The BI-VAR formulation also includes two families of clique constraints, formulated over the binary variables, which have analogous constraints in both the TI and BI models. They are

$$\sum_{b \in B_j} \sum_{b' \in B^c_{jb}} z_{jbb'} = 1, \qquad j \in J, \tag{14}$$
$$\sum_{j \in J} \sum_{\substack{b \in B_j: \\ b \leq a}} \sum_{\substack{b' \in B^c_{jb}: \\ b' > a}} z_{jbb'} \leq 1, \qquad a \in B. \tag{15}$$

Constraints (14) ensure that each job is processed exactly once and for each bucket $a \in B$, constraints (15) ensure that at most one job can be processed in both bucket a and bucket $a + 1$. In general, these constraints are not sufficient to ensure that the machine capacity is not violated as they do not prevent adjacent jobs

from overlapping when the overlap is entirely contained within a bucket. We therefore need the following constraints, each of which is specific to one of the three classes of buckets described in Section 3. We note that every bucket in B^2 could alternatively belong to B^3, and furthermore could also satisfy the conditions for membership in B^∞. If a bucket could belong to more than one of these sets preference is given in the order B^2, B^3, B^∞, which we briefly motivate at the end of Section 4.3.

4.1 At most two jobs per bucket (B^2)

The following constraints are analogous to constraints in the BI model, and along with constraints (15), ensure the capacity of the machine is respected for every bucket $b \in B^2$:

$$\sum_{j \in J} \Big(\sum_{b'' \in B^c_{jb}} u_{jbb''} + \sum_{\substack{b' \in B_j: \\ b \in B^c_{jb'}}} v_{jb'b} + \sum_{b' \in B_j: b' < b} \sum_{\substack{b'' \in B^c_{jb'}: \\ b'' > b}} z_{jb'b''} \Big) \leq 1, \ b \in B^2. \tag{16}$$

4.2 At most three jobs per bucket (B^3)

The following three constraints, together with constraints (15), ensure the capacity of the machine is respected for each bucket $b \in B^3$. The first constraint ensures that the sum of all processing time incurred in bucket b is less than the length of the bucket.

$$\sum_{j \in J_b} p_j / \Delta_b z_{jbb} + \sum_{j \in J} \Big(\sum_{b'' \in B^c_{jb} \setminus \{b\}} u_{jbb'} + \sum_{\substack{b' \in B_j \setminus \{b\}: \\ b \in B^c_{jb'}}} v_{jb'b} + \sum_{b' \in B_j: b' < b} \sum_{\substack{b'' \in B^c_{jb'}: \\ b'' > b}} z_{jb'b''} \Big) \leq 1, \ b \in B^3. \tag{17}$$

While the above constraints ensure that the amount of processing scheduled for a bucket $b \in B^3$ is valid, it does not prevent a job which both starts and completes in bucket b from overlapping the preceding or succeeding job. The following two families of constraints achieve this, respectively:

$$\sum_{j \in J_b} u_{jbb} + \sum_{j \in J} \Big(\sum_{\substack{b' \in B_j \setminus \{b\}: \\ b \in B^c_{jb'}}} v_{jb'b} + \sum_{b' \in B_j: b' < b} \sum_{\substack{b'' \in B^c_{jb'}: \\ b'' > b}} z_{jb'b''} \Big) \leq 1, \ b \in B^3 \tag{18}$$

$$\sum_{j \in J_b} v_{jbb} + \sum_{j \in J} \Big(\sum_{b'' \in B^c_{jb} \setminus \{b\}} u_{jbb''} + \sum_{b' \in B_j: b' < b} \sum_{\substack{b'' \in B^c_{jb'}: \\ b'' > b}} z_{jb'b''} \Big) \leq 1 \ b \in B^3. \tag{19}$$

4.3 To infinity and beyond (B^∞)

We now use a result which we attribute to Sadykov (2006) to show that under certain conditions a model can be formulated to include buckets which allow any number of jobs to be scheduled. Let $b \in B$ and once again define J_b as above. If for each $j \in J_b$ the open interval $(t_{b-1}, t_b + p_j)$ does not contain the release date, due date or deadline for job j, then each of these jobs can be feasibly started at any time in bucket b and J_b can be separated into *early* jobs and *late* jobs. Early jobs, if started and completed in bucket b, begin prior to their due date, and late jobs, if started and completed in bucket b, begin on or after their due date. Sadykov recognised that if a feasible solution for the SMSP instance exists, then there is an optimal schedule in which those jobs that start and complete in bucket b, are ordered so that late jobs precede early jobs, and the late jobs are sequenced according to the well known *Smith's rule* (Smith (1956)). The a priori sequence of early jobs is arbitrary since they incur no cost. Formally, we define an order σ_b on the set J_b, such that

(i) if $i, j \in J_b$ then $\sigma_b(i) < \sigma_b(j)$ if job i is late and job j is early, and

(ii) if $i, j \in J_b$ and both jobs are late then $\sigma_b(i) < \sigma_b(j) \Rightarrow \frac{w_i}{p_i} \geq \frac{w_j}{p_j}$.

For each $j \in J_b$ we also define $A^-_{jb} = \{i \in J_b : \sigma_b(i) < \sigma_b(j)\}$ and $A^+_{jb} = \{i \in J_b : \sigma_b(j) < \sigma_b(i)\}$. Then if jobs $i, j \in J_b$ both start and complete in bucket b and $i \in A^-_{jb}$, then we require i to be processed prior to j. We claim the following constraints, along with constraints (15) and (17) ensure the capacity of the machine is respected for each bucket $b \in B^\infty$. The first family of constraints are analogous to (18) as, for

bucket $b \in B^\infty$, they prevent any job $i \in J_b$ which both starts and completes in bucket b from overlapping any job which precedes it.

$$u_{ibb} + \sum_{j \in A^-_{ib}} p_j/\Delta_b z_{jbb} + \sum_{j \in J} (\sum_{\substack{b' \in B_j \setminus \{b\}: \\ b \in B^c_{jb'}}} v_{jb'b} + \sum_{\substack{b' \in B_j: \\ b' < b}} \sum_{\substack{b'' \in B^c_{jb'}: \\ b'' > b}} z_{jb'b''}) \leq 1, \ b \in B^\infty, i \in J_b, \tag{20}$$

Similarly the constraints below are analogous to (19) as, for bucket $b \in B^\infty$, they prevent any job $i \in J_b$ which both starts and completes in bucket b from overlapping any job which succeeds it.

$$v_{ibb} + \sum_{j \in A^+_{ib}} p_j/\Delta_b z_{jbb} + \sum_{j \in J} (\sum_{b' \in B^c_{jb} \setminus \{b\}} u_{jbb'} + \sum_{\substack{b' \in B_j: \\ b' < b}} \sum_{\substack{b'' \in B^c_{jb}: \\ b'' > b}} z_{jb'b''}) \leq 1, \ b \in B^\infty, i \in J_b. \tag{21}$$

If a bucket b can belong to B^2 and either B^3 or B^∞ then $J_b = \emptyset$ and constraints (17) are equivalent to (16). Furthermore if b can belong to B^3 then constraints (18) and (19) are both dominated by (17), and therefore also by (16). If a bucket b can belong to B^3 or B^∞ then constraints (20) and (21) are dominated by (18) and (19), respectively. A proof of these claims can be found in Clement (2013).

4.4 The complete BI-VAR formulation

The entire formulation for the BI-VAR model is then given by constraints (13)–(21) as well as the following domain constraints on the variables:

$$u_{jbb'} \leq u^u_{jbb'} z_{jbb'}, \qquad j \in J, b \in B_j, b' \in B^c_{jb}, \tag{22}$$
$$u_{jbb'} \geq u^l_{jbb'} z_{jbb'}, \qquad j \in J, b \in B_j, b' \in B^c_{jb}, \tag{23}$$
$$v_{jbb'} \leq v^u_{jbb'} z_{jbb'}, \qquad j \in J, b \in B_j, b' \in B^c_{jb}, \tag{24}$$
$$v_{jbb'} \geq v^l_{jbb'} z_{jbb'}, \qquad j \in J, b \in B_j, b' \in B^c_{jb}, \tag{25}$$
$$z_{jbb'} \in \{0, 1\}, \qquad j \in J, b \in B_j, b' \in B^c_{jb}. \tag{26}$$

Constraints (22)–(25) bound the continuous variables, such that if the corresponding binary variable, say $z_{jbb'}$, is nonzero then $s_j \in I_b$ and $c_j \in I_{b'}$ where s_j and c_j are defined by equations (11) and (12) respectively.

Omitted from this model are constraints which enforce that jobs respect their release dates and due dates. We claim that these restrictions are easily incorporated into the model by adding inequalities similar to constraints (22) and (23), which would bound the u variables, which in turn bound the start times of jobs in feasible schedules. We refer the reader to Boland et al. (2013) or Clement (2013) for a detailed explanation of how release dates and deadlines, in addition to the objective function, are modelled in the BI formulation as the approach used for the BI-VAR formulation is equivalent.

5 RELATION TO THE BI MODEL

If the all the buckets are of uniform length then the variable definitions are equivalent, although not explicitly identical, to those presented in the BI model (Boland et al. (2013)). The variables in the BI model are indexed by the sets J, B, and K, where $K = \{0, 1\}$. A nonzero value for a binary variable z_{jbk} in the BI model indicates that job j starts in bucket b and concludes in bucket $b + P_j - k + 1$ where P_j is a constant calculated from the input data. When all buckets are the same length in the BI-VAR model, we can use the equation $k = b + P_j + 1 - b'$ to infer that variables z_{jbk} and $z_{jbb'}$ from the BI and BI-VAR formulations respectively, are semantically identical. If the buckets are of equal size in the BI-VAR model the constraints are also equivalent to those in the BI model. A proof of these claims can be found in Clement (2013).

6 CONCLUSIONS

In this paper we present the variable sized bucket indexed (BI-VAR) formulation, a mixed integer linear program for solving nonpreemptive single machine scheduling problems (SMSP). The BI-VAR model is a generalisation of the bucket indexed (BI) formulation, proposed by the same authors, which itself generalises the classical time indexed (TI) model. Like the BI model, the BI-VAR formulation requires the time horizon to be partitioned into time intervals referred to as buckets, however unlike the BI model, the BI-VAR formulation

permits partitions which induce nonuniform bucket sizes. The freedom of choice for the partition used in the BI-VAR formulation raises interesting questions of how to best partition the time horizon given a particular problem input. We show in our previous study how a particular family of facet defining inequalities for the TI model can be mapped into valid inequalities for the BI model and under certain conditions remain facet defining. We expect a similar result for the BI-VAR model can be obtained. Other potential investigations include network formulations analogous to the arc time indexed formulation (Tanaka et al. (2009), Sourd (2009), Pessoa et al. (2010)) which is an extended formulation for the TI model. We conclude by acknowledging that we expect that the BI-VAR will be able to solve instances of the SMSP which are intractable for both TI and BI models, however a comprehensive computational study comparing these models is required.

REFERENCES

Baptiste, P. and R. Sadykov (2009). On scheduling a single machine to minimize a piecewise linear objective function: A compact mip formulation. *Naval Research Logistics (NRL) 56*(6), 487–502.

Boland, N., R. Clement, and H. Waterer (2013). A big bucket time indexed formulation for nonpreemptive single machine scheduling problems. Report C-OPT 2013-002, Centre for Optimal Planning and Operations, University of Newcastle, Australia. `hdl.handle.net/1959.13/941091`.

Brucker, P. (2007). *Scheduling Algorithms*. Springer.

Clement, R. (2013). *Mixed Integer Programming Formulations for Sequencing and Scheduling with an Application to a Coal Loading Facility*. Ph. D. thesis, University of Newcastle, Australia. In preparation.

Dyer, M. E. and L. A. Wolsey (1990). Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics 26*(23), 255 – 270.

Hall, L., A. Schulz, D. Shmoys, and J. Wein (1997). Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Mathematics of Operations Research 22*, 513–544.

Khowala, K., A. Keha, and J. Fowler (2005). A comparison of different formulations for the non-preemptive single machine total weighted tardiness scheduling problem. In G. Kendall, L. Lei, and M. Pinedo (Eds.), *In proceedings of the 2nd Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2005), 18 -21 July 2005, New York, USA*, pp. 643–651. Paper.

Lawler, E., J. Lenstra, A. Rinnooy Kan, and D. Shmoys (1993). Sequencing and scheduling: Algorithms and complexity. In S. Graves, A. Rinnooy Kan, and P. Zipkin (Eds.), *Logistics of Production and Inventory*, Volume 4 of *Handbooks in Operations Research and Management Science*, pp. 445–522. North Holland.

Pessoa, A., E. Uchoa, M. Poggi de Aragão, and R. Rodrigues (2010). Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation 2*, 259–290.

Pinedo, M. (2002). *Scheduling: theory, algorithms, and systems.* Prentice Hall.

Pochet, Y. and L. Wolsey (2006). *Production Planning by Mixed Integer Programming*. Springer Series in Operations Research and Financial Engineering. Springer.

Sadykov, R. (2006). *Integer programming-based decomposition approaches for solving machine scheduling problems*. Ph. D. thesis, Université Catholique de Louvain, Belgium.

Smith, W. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly 3*(1-2), 59–66.

Sourd, F. (2009). New exact algorithms for one-machine earliness-tardiness scheduling. *INFORMS Journal on Computing 21*(1), 167–175.

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

Tanaka, S., S. Fujikuma, and M. Araki (2009). An exact algorithm for single-machine scheduling without machine idle time. *Journal of Scheduling 12*(6), 575–593.

Van den Akker, J., C. Van Hoesel, and M. W. P. Savelsbergh (1999). A polyhedral approach to single-machine scheduling problems. *Mathematical Programming 85*(3), 541–572.