Scheduling Projects with Labour Constraints

C. C. de Souza *L. A. Wolsey*

Relatório Técnico IC–97-22

Novembro de 1997

Scheduling Projects with Labour Constraints †

C. C. de Souza‡ L. A. Wolsey§

Instituto de Computação
Universidade Estadual de Campinas — UNICAMP
Caixa Postal 6176 – CEP: 13083-970 – Campinas, SP – Brasil

October 1997

Abstract. In this paper we consider a labour constrained scheduling problem which is a simplification of a practical problem arising in the industry. Jobs are subject to precedence constraints and have specified processing times. Moreover, for each job the labour requirement varies as the job is processed. Given the amount of labour available in each period, the problem is to finish all the jobs as soon as possible (minimise *makespan*, subject to the precedence and labour constraints). Several Integer Programming (IP) formulations for this problem are discussed and valid inequalities for these different models are introduced. We point out to the major drawbacks in using the IP approach which are essentially due to the weakness of the lower bound relaxations. However, we report computational experiments showing how IP can be used to provide good feasible solutions and we indicate directions for further investigations which may turn IP techniques an interesting tool for solving such a problem.

Keywords: *Scheduling, Labour Constraints, Integer Programming, Valid Inequalities, Heuristics.*

1 Introduction

The labour constrained scheduling problem involves sequencing a set of jobs subject to precedence constraints represented by a digraph where each job has a labour profile. Thus each job has a specified processing time, and the labour requirement varies as the job is processed. Given the amount of labour available in each period, the problem is to finish all the jobs as soon as possible (minimise *makespan*, subject to the precedence and labour constraints).

The problem motivating this study appears in Heipcke [7] and is a simplification of an industrial problem from BASF. The instances we study are tightly constrained. For these instances Cavalcante and de Souza [2] have developed tabu search heuristics, Heipcke has

†This research was supported by FAPESP (grant number 97/02990-3) and by CNPq (grant number 300883/94-3)

‡Institute of Computing – State University of Campinas – São Paulo – Brazil

§Center for Operations Research and Econometrics – UCL – Louvain la Neuve – Belgium

developed a constraint logic approach [8] and Savelsbergh, Wang and Wolsey [10] have developed a heuristic integer programming approach, based on the order of the jobs suggested by the linear programming solution. This idea appears in [12], and has recently been the subject of several worst case heuristic analyses [6],[9]. Other than using the longest path lower bound just based on the processing times and precedence digraph, these approaches provide no guarantee of the quality of the solution found. Wang and Wolsey [13] have studied integer programming formulations and cutting planes in an attempt to improve these lower bounds, but their results are limited.
Therefore, a major motivation for studying integer programming approaches to the problem is to provide performance guarantees (lower bounds on the optimal makespan) as well as good feasible schedules (upper bounds on the optimal makespan).
To date, the approach via Integer Programming cannot be judged a success for the following reasons:

(i) in spite of the development of several alternative formulations for the problem, the weak lower bounds provided by the longest path is not significantly improved.

(ii) one 24 job instance has been solved to optimality using strong valid inequalities. However, the inequalities are very instance specific and appear difficult to generalize.

(iii) the systems used to solve the integer programs have great difficulty in finding feasible solutions when the instances are tightly constrained. The quality of the solutions that are found is worse than those provided by the tabu, CLP heuristics and LP order heuristics.

(iv) the improvements in the lower bound that can be obtained using IP are small, and it appears likely that the heuristic solutions are close to optimal, whereas the lower bounds are not.

Below we discuss the previous and our more recent attempt to tackle the problem using integer programming. In Section 2 we discuss formulations. One feature of the test instances is that there exist chains of identical jobs. Formulations are designed precisely to treat such instances.
In Section 3 we discuss known valid inequalities for the problem. Those of Wang and Wolsey are presented as well as some that can be obtained from partial relaxations of a problem instance.
In Section 4 we discuss various nonstandard ways in which the IP formulation can be used both to improve on the weak LP lower bound and to find feasible solutions.
The reasons for the relative weakness of general IP approach are discussed in Section 5, and some ideas for the construction of a special purpose branch-and-bound code are presented.

2 Formulations

Because of the labour requirements, any explicit integer programming formulation requires knowledge of the period in which each job starts. The data consists of a set $N = \{1, \ldots, n\}$

of jobs, a digraph $D = (N, A)$ representing the precedence constraints, the processing time p_j and labour profile $(\ell_{j,1}, \ell_{j,2}, \ldots, \ell_{j,p_j})$ of each job $j \in N$. The labour capacity is L in each period. A time horizon T must be chosen. Periods are $1, 2, \ldots, T$ representing the intervals $[0,1], [1,2], \ldots, [T-1, T]$.
In this section we discuss various different formulations for the processing times, precedence constraints and makespan objective, that are independent of the labour constraints.

2.1 The basic time-indexed start-time x-formulation

Let ξ denote the objective value (makespan+1), μ an upper bound on the makespan, $x_{j,t} = 1$ if job j starts in period t and s_j the start time of job j.
Now the problem can be formulated as:

$$\begin{aligned}
\xi &= 1 + \min \mu && (1)\\
\mu &\geq s_j + p_j \quad j \in N, && (2)\\
\sum_{t=1}^{T} x_{j,t} &= 1 \quad j \in N, && (3)\\
\sum_{t=1}^{T} t x_{j,t} &= s_j \quad j \in N, && (4)\\
s_j &\geq s_i + p_i \quad (i,j) \in A, && (5)\\
\sum_{s=1}^{t} x_{i,s} &\geq \sum_{s=1}^{t+p_i} x_{j,s} \quad (i,j) \in A, t \in [1, \ldots, T], && (6)\\
\sum_{j=1}^{n} \sum_{u=1}^{p_j} \ell_{j,u} x_{j,t-u+1} &\leq L \quad t \in [1, \ldots, T], && (7)\\
x_{j,t} &\in \{0,1\} \qquad j \in N, t \in [1, \ldots, T]. && (8)
\end{aligned}$$

Constraint (2) imposes that μ is an upper bound on the makespan, constraint (3) that each job j is carried out, and (4) links the s_j and $x_{j,t}$ variables. Clearly the s_j variables can be eliminated by substitution. Constraint (5) is a simple representation of the precedence constraints, whereas (6) is a tighter representation involving a large number of constraints. Finally (7) is the labour constraint.
To minimise makespan, one approach is to introduce a final dummy job which starts in the period when all the real jobs have terminated. From now on, we assume that the definition of N and D have been modified, the dummy job is a successor of all real jobs and it is job n.
The formulation now becomes:

$$\xi = \min s_n, \text{ subject to, } (3), (4), (5) \text{ or } (6), (7) \text{ and } (8).$$

Note also that the labour constraint also can be modified giving:

$$\sum_{j=1}^{n-1}\sum_{u=1}^{p_j} \ell_{j,u} x_{j,t-u+1} + L\sum_{s=1}^{t} x_{n,s} \leq L, t \in [1,\ldots,T]. \tag{9}$$

Also based on the precedence constraints and the time horizon T, it is easy to calculate earliest and latest start times for each job $j \in N$ by forward and backward longest path calculations in the digraph D using the processing times p_j, $j \in N$. Variables $x_{j,t}$ are then only defined for $t = e(j), e(j)+1, \ldots, f(j)$, and all the constraints are modified appropriately. Thus, ignoring the labour constraints, we are interested in formulations for the feasible region $S^x = \{(x,s) : (3),(4),(5) \text{ or } (6),(8)\}$ for which we have two possible formulations, the "weak"formulation: $Q_1^x = \{(x,s) : (3),(4),(5), 0 \leq x_{j,t} \leq 1, e(j) \leq t \leq f(j), j \in N\}$ and the "strong" one: $Q_2^x = \{(x,s) : (3),(4),(6), 0 \leq x_{j,t} \leq 1, e(j) \leq t \leq f(j), j \in N\}$.
The term strong is justified by the following result that will be proved in the next section and says that all extreme points of Q_2^x are integral.

Proposition 1 *Q_2^x is integral.*

The proposition also tells us that every valid inequality for S^x is a nonnegative linear combination of inequalities describing Q_2^x.

Proposition 2 *Consider a sequence of jobs $j_1, j_2, \ldots, j_r \in N$ with $(j_i, j_{i+1}) \in A$ for $i = 1,\ldots,r-1$. For all $t_1 \geq t_2 \geq \ldots \geq t_r$, with $e(j_i) \leq t_i \leq f(j_i)$, $i = 1,\ldots,r-1$,*

$$\sum_{i=1}^{r}\sum_{u=1}^{p_{j_i}} x_{j_i,t_i-u+1} \leq 1 \tag{10}$$

is valid for S^x.

Again the proof will be simpler with the notation of the following subsection. The *path inequalities* (10) are redundant for Q_2^x, but may be useful perhaps with $t = t_1 = t_2 = \ldots = t_n$ to tighten formulation Q_1^x without introducing too many new constraints.

2.2 The cumulative start time z-formulation

A standard time-indexed variable reformulation involves the replacement of the start-time variables $x_{j,t}$ by the cumulative variables $z_{j,t}$. Specifically we define: $z_{j,t} = 1$ if job j starts in or before period t.
Clearly,

$$z_{j,t} = \sum_{s=e(j)}^{t} x_{j,t}, \text{for} \quad t = e(j),\ldots,f(j),$$

and conversely $x_{j,e(j)} = z_{j,e(j)}$, and $x_{j,t} = z_{j,t} - z_{j,t-1}$ for $t = e(j)+1,\ldots,f(j)$. Now constraint (3) becomes:

$$z_{j,f(j)} = 1 \qquad j \in N,$$

and from $x_{j,t} \geq 0$ we get

$$0 \leq z_{j,e(j)} \leq \ldots \leq z_{j,f(j)}$$

with $z_{j,t} \in \{0,1\}$. The precedence constraints (6) become

$$z_{i,t} \geq z_{j,t+p_i}$$

and one gets,

$$s_j = \sum_{t=e(j)}^{f(j)} t x_{j,t} = \sum_{t=e(j)}^{f(j)} t(z_{j,t} - z_{j,t-1}) = f(j) + 1 - \sum_{t=e(j)}^{f(j)} z_{j,t}.$$

Thus, we immediately obtain the set S^z and the formulations Q_1^z, Q_2^z in the z-space corresponding to S^x, Q_1^x and Q_2^x respectively.

$$\begin{aligned}
Q_2^z &= \{(s,z): \\
s_j &= f(j) + 1 - \sum_{u=e(j)}^{f(j)} z_{j,u} \quad j \in N, && (11)\\
z_{j,f(j)} &= 1 \quad j \in N, && (12)\\
z_{j,t-1} - z_{j,t} &\leq 0 \quad j \in N, t = e(j)+1, \ldots, f(j), && (13)\\
-z_{j,e(j)} &\leq 0 \quad j \in N, && (14)\\
z_{i,t} - z_{j,t+p_i} &\geq 0 \quad (i,j) \in A, e(i) \leq t \leq f(i), e(j) \leq t + p_i \leq f(j)\} && (15)
\end{aligned}$$

If (15) is replaced by (5) in the definition of Q_2^z we obtain Q_1^z. Note that $S^z = \{(z,s) \in Q_1^z : z \text{ integer}\}$.

Proposition 3 *The polyhedron Q_2^z is integral.*

Proof: The matrix described by (12), (13), (14) and (15) is totally unimodular as each constraint contains either a single entry, or two nonzero entries that are $+1$ and -1. □

As there is a bijection between Q_2^x and Q_2^z, Proposition 1 follows.

Proposition 4 (11) *Subject to the conditions of Proposition 2, the path inequality*

$$\sum_{i=1}^{r} (z_{j_i,t_i} - z_{j_i,t_i-p_{j_i}}) \leq 1 \tag{16}$$

is valid for S^z.

Proof: The LHS of the inequality can be written as:

$$\begin{aligned}
&z_{j_1,t_1} - \textstyle\sum_{i=1}^{r-1} (z_{j_i,t_i-p_{j_i}} - z_{j_{i+1},t_{i+1}}) - z_{j_r,t_r-p_{j_r}} && = \\
&z_{j_1,t_1} - \textstyle\sum_{i=1}^{r-1} (z_{j_i,t_i-p_{j_i}} - z_{j_{i+1},t_i}) - \sum_{i=1}^{r-1} (z_{j_{i+1},t_i} - z_{j_{i+1},t_{i+1}}) - z_{j_r,t_r-p_{j_r}} && \leq 1
\end{aligned}$$

as $z_{j_1,t_1} \leq 1$ by (12) and (13), $z_{j_i,t_i-p_{j_i}} - z_{j_{i+1},t_i} \geq 0$ by (15), $z_{j_{i+1},t_i} - z_{j_{i+1},t_{i+1}} \geq 0$ by (13) as $t_i \geq t_{i+1}$, and $z_{j_r,t_r-p_{j_r}} \geq 0$ by (13) and (14). □

Again Proposition 2 follows immediately.

2.3 The Block Formulation

A block B is a sequence of n_B identical jobs $j_1, \ldots, j_{n_B}$ with $(j_i, j_{i+1}) \in A$, $i = 1, \ldots, n_B - 1$ of the same length and same labour profile. We abuse notation by using $e(B) = e(j_1)$, $f(B) = f(j_{n_B})$ for the earliest and latest times for some job in the block to start, and $p_B = p_{j_1}$, $\ell_{B,u} = \ell_{j_1,u}$ for the processing times and labour profiles of the jobs in block B.
Throughout this section we assume that the problem is modelled with blocks such that any precedence constraint $(i, j) \in A$ between job i in a block B and job j in a block B', $B \neq B'$ is such that i is the last job in block B and j is the first job in block B'. Thus we obtain a block precedence graph $\overline{D}$ with arc set given by $\overline{A}$. Figure 1 gives an example of the notation we use.

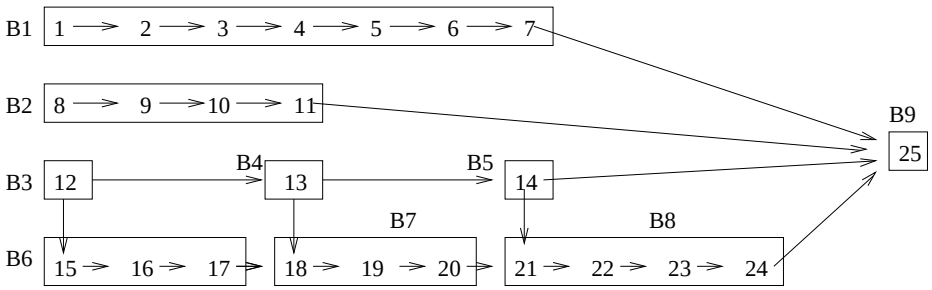


Figure 1: Example of blocks

Define a block variable $X_{B,t} = \sum_{j \in B} x_{j,t}$ so that $X_{B,t} = 1$ if some job of block B starts at time t.
To obtain a representation of the feasible region in the X-space, we obtain

$$\sum_{t=e(B)}^{f(B)} X_{B,t} = n_B, \qquad \forall B \in \mathcal{B} \tag{17}$$

$$X_{B,t} \in \{0, 1\} \qquad \forall B \in \mathcal{B} \;\; t = 1, \ldots, T, \tag{18}$$

from (3) and (8) respectively, where $\mathcal{B}$ is the set of all blocks.
The labour constraint (7) becomes

$$\sum_{B \in \mathcal{B}} \sum_{u=1}^{p_B} \ell_{B,u} X_{B,t-u+1} \leq L. \tag{19}$$

To model the precedence constraints is less obvious. The path inequality (10) with $t = t_1, t_2, \ldots, t_{n_B}$ when limited to jobs in block B becomes

$$\sum_{u=1}^{p_B} X_{B,t-u+1} \leq 1. \tag{20}$$

To describe other precedence constraints it is simplest to first introduce cumulative block variables. For this, let $Z_{B,t} = \sum_{j \in B} z_{j,t}$ be the number of jobs of block B that have started

in or before period t. Note that $Z_{B,t}$ is also the cumulative variable for the block start time variable $X_{B,t}$ so that

$$Z_{B,t} = \sum_{s=e(B)}^{t} X_{B,s}, \qquad t = e(B), \ldots, f(B).$$

In the absence of precedence constraints between blocks, constraints (17), (18) and (20) now lead to the formulation R^z:

$$\begin{array}{lcl} Z_{B,f(B)} & = & n_B \\ Z_{B,t} - Z_{B,t+1} & \leq & 0 \\ -Z_{B,e(B)} & \leq & 0 \\ Z_{B,t} - Z_{B,t-p_B} & \leq & 1. \end{array}$$

Proposition 5 *The formulation R^z is integral.*

Proof: As in the proof of Proposition 1, the matrix is totally unimodular. □

Theoretically, at least we also can find the convex hull of the set S^Z of feasible solutions to the block formulation in the presence of precedence constraints between blocks.

$$\begin{array}{lcll} \text{Let} \quad \Phi \quad = \quad \{(z,Z) : \; z_{i,(f_i)} & = & 1 & i = 1, \ldots, n \\ z_{i,t} - z_{i,t+1} & \leq & 0 & i = 1, \ldots, n,\ t = 1, \ldots, T-1 \\ -z_{i,e(i)} & \leq & 0 & i = 1, \ldots, n \\ -z_{i,t} + z_{j,t+p_i} & \leq & 0 & \forall (i,j) \in A,\ t = 1, \ldots, T - p_i \\ \sum_{j \in B} z_{j,t} & = & Z_{B,t} & \forall B \in \mathcal{B},\ t = e(B), \ldots, f(B).\} \end{array}$$

Proposition 6 $Proj_Z(\Phi) = conv(S^Z)$.

Proof: This is an immediate corollary of the integrality of Q_2^z. □

We now derive inequalities to represent precedence constraints between blocks. For a path of blocks, $B_1, B_2, \ldots, B_r$ with $(B_i, B_{i+1}) \in \bar{A}$ for $i = 1, \ldots, r$, we immediately obtain a path block inequality from Proposition 4 and (20).

Proposition 7 (11) *The inequality*

$$\sum_{i=1}^{r} (Z_{B_i,t_i} - Z_{B_i,t-p_{B_i}}) \leq 1, \tag{21}$$

with $t_1 \geq t_2 \geq \ldots \geq t_r$ is valid for S^Z.

Proposition 8 *Let B and C be two blocks with $(B,C) \in \overline{A}$. The inequality*

$$\sum_{u=1}^{n_C} Z_{B,t+(u-1)p_C} \geq \sum_{u=1}^{n_B} Z_{C,t+up_B+(n_C-1)p_C} \tag{22}$$

is valid for S^Z.

Proof: Let $B = \{b_1, \ldots, b_{n_B}\}$ and $C = \{c_1, \ldots, c_{n_C}\}$. The precedence constraints between jobs imply that for $1 \le u \le n_B$ and $1 \le s \le n_C$,

$$\begin{array}{rcccccl} z_{c_s,t+up_B+(n_C-1)p_C} & \le & z_{c_{s-1},t+up_B+(n_C-2)p_C} & \le & \ldots & \le & z_{c_1,t+up_B+(n_C-s)p_C} \\ & \le & z_{b_{n_B},t+(u-1)p_B+(n_C-s)p_C} & \le & \ldots & \le & z_{b_{n_B-u+1},t+(n_C-s)p_C} \end{array}$$

Summing over $u = 1, \ldots, n_B$ gives

$$\sum_{u=1}^{n_B} z_{c_s,t+up_B+(n_C-1)p_C} \le \sum_{u=1}^{n_B} z_{b_{n_B-u+1},t+(n_C-s)p_C = Z_{B,t+(n_C-s)p_C}}.$$

Now summing over $s = 1, \ldots, n_C$ and changing the order of summation gives

$$\sum_{u=1}^{n_B}\sum_{s=1}^{n_C} z_{c_s,t+up_B+(n_C-1)p_C} = \sum_{u=1}^{n_B} Z_{C,t+up_B+(n_C-1)p_C} \le \sum_{s=1}^{n_C} Z_{B,t+(n_C-s)p_C} = \sum_{u=1}^{n_C} Z_{B,t+(u-1)p_C}.$$

□

In the special case when either $|B| = 1$, or $|C| = 1$, this result appears in [12].

2.4 The block-job formulation

An alternative to reducing the size of blocks so as to just impose the precedence constraints between blocks is to make blocks as large as possible, but retain every job involved in a precedence constraint between two blocks. Such jobs are called *special*.
Again we have the $x_{j,t}$, $z_{j,t}$ variables for the special jobs and $X_{B,t}$ and $Z_{B,t}$ variables for the blocks.
All precedence constraints between blocks can be handled by the special jobs, so for $(i, j) \in A$ with i and j in different blocks, we have:

$$z_{i,t} \ge z_{j,t+p_i}, \qquad t = 1, \ldots, T.$$

Precedence constraints between special jobs in the same block can be handled as follows: Let $B = \{1, \ldots, n_B\}$ be a block with special jobs $j_1, \ldots, j_r$ in positions $1 \le q_{j_1} < \ldots < q_{j_r} \le n_B$. For any two consecutive special jobs in B, j_ℓ, $j_{\ell+1}$, $\ell = 1, \ldots, r-1$, the precedence inequality below holds:

$$z_{j_\ell,t} \ge z_{j_{\ell+1},t+(q_{j_{\ell+1}}-q_{j_\ell})p_B} \qquad t = 1, \ldots, T.$$

Proposition 9 *If block $B = \{1, \ldots, n_B\}$ contains special jobs $j_1, \ldots, j_r$ in positions $1 \le q_{j_1} < \ldots < q_{j_r} \le n_B$, then*

$$Z_{B,t} \ge \sum_{u=1}^{q_{j_1}} z_{j_1,t+(u-1)p_B} + \sum_{u=1}^{q_{j_2}-q_{j_1}} z_{j_2,t+(u-1)p_B} + \ldots + \sum_{u=1}^{q_{j_r}-q_{j_{r-1}}} z_{j_r,t+(u-1)p_B}, \tag{23}$$

$$Z_{B,t} \le (q_{j_1} - 1) + \sum_{v=1}^{r} \sum_{u=1}^{q_{j_{v+1}}-q_{j_v}} z_{j_v,t-(u-1)p_B}. \tag{24}$$

with $q_{j_{r+1}} = n_B + 1$.

Proof:

(i)

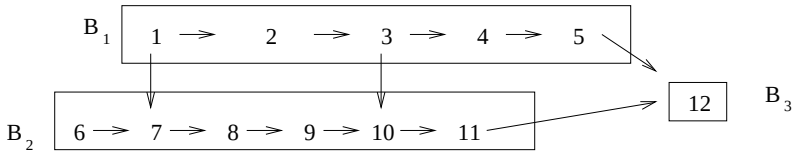
$$\begin{aligned} Z_{B,t} &= (z_{1,t} + z_{2,t} + \ldots + z_{j_1,t}) + (z_{j_1+1,t} + \ldots + z_{j_2,t}) + \ldots \\ &\quad +(z_{j_r+1,t} + \ldots + z_{n_B,t}) \\ &\geq z_{j_1,t+(q_{j_1}-1)p_B} + z_{j_1,t+(q_{j_1}-2)p_B} + \ldots + z_{j_1,t} \\ &+ z_{j_2,t+(q_{j_2}-q_{j_1}-1)p_B} + z_{j_2,t+(q_{j_2}-q_{j_1}-2)p_B} + \ldots + z_{j_2,t} \\ &+ \ldots \\ &+ 0 \end{aligned}$$

(ii)

$$\begin{aligned} Z_{B,t} &= (z_{1,t} + z_{2,t} + \ldots + z_{j_1-1,t}) + (z_{j_1,t} + \ldots + z_{j_2-1,t}) + \ldots \\ &\leq (q_{j_1} - 1) + (z_{j_1,t} + z_{j_1,t-p_B} + \ldots + z_{j_1,t-(q_{j_2}-q_{j_1})p_B}) \\ &+ \ldots \end{aligned}$$

□

An example of the previous inequality is given below.
Example:



There are three blocks: B_1 with jobs from 1 to 5, B_2 with jobs from 6 to 11 and B_3 with job 12 (dummy). The special jobs are 1, 3, 5, 7, 10, 11 and 12.
For B_2 we obtain the inequality:

$$Z_{B_2,t} \geq (z_{7,t+p_{B_2}} + z_{7,t}) + (z_{10,t+2p_{B_2}} + z_{10,t+p_{B_2}} + z_{10,t}) + (z_{11,t})$$

and

$$Z_{B_2,t} \leq 1 + (z_{7,t} + z_{7,t-p_{B_2}} + z_{7,t-2p_{B_2}}) + (z_{10,t}) + (z_{11,t}).$$

□

Note that path inequalities can again be applied to the blocks.

3 Labour Constraints and Inequalities

3.1 Knapsack inequalities for blocks (or jobs)

The labour constraints (19) with the path constraints (20) for each block form together a knapsack with GUB (generalized upper bound) structure. Let $(B'_i, u'_i)_{i=1}^r$, $B'_i \in \mathcal{B}$, $1 \leq u \leq p_{B'_i}$ be a GUB cover with the $\{B'_i\}_{i=1}^r$ distinct and $\sum_{i=1}^r \ell_{B'_i,u'_i} > L$.

Proposition 10 *The GUB cover inequality*

$$\sum_{i=1}^{r} \sum_{u:\ell_{B_i,u} \geq \ell_{B'_i,u'_i}} X_{B_i,t-u+1} + \sum_{B \notin \{B_1,\ldots,B_r\}:\ell_{B,u} \geq \max_i \{\ell_{B'_i,u'_i}\}} \alpha_{B,u} X_{B,t-u+1} \leq r-1 \qquad (25)$$

is valid for the block model with $\alpha_{B,u} = 1$.

Note that the blocks can be replaced by disjoint block paths.

Corollary 1 *If* $\ell_{B,u} = L$ *for some* (B,u) *and* $\ell_{B',u'} > 0$, *then* (B,u) *and* (B',u') *form a cover. Hence for any path* P *not containing* B

$$\sum_{B' \in P} \sum_{u':\ell_{B',u}>0} X_{B',t-u'+1} + X_{B,t-u+1} \leq 1. \qquad (26)$$

The next inequality for job or block models uses much more of the problem structure, but requires very strict conditions.
We consider $k \geq 3$ jobs such that the following hold for some values α, β, ξ, t^*:

1. $\xi = \min_{j,u}\{\ell_{j,u}\} > 0$;
2. $\ell_{k,u} \geq \beta$ for some $1 \leq u \leq p_k$;
3. For $i = 1,\ldots,k-1$, $\exists$ integers $h_i \leq t^*$ such that $\ell_{i,u} \geq \alpha$ for $u = 1,\ldots,h_i$;
4. $2\alpha > L$, $\alpha + \beta \leq L$, $\alpha + \beta + \xi > L$.

Proposition 11 *[13] For any integer* q^* *with* $t^* \leq q^* \leq \min\{p_j : j = 1,\ldots,k-1\}$, *the inequality*

$$\sum_{i=1}^{k} \sum_{u=1}^{\min\{p_i,q^*+h_i\}} x_{j,t-u+1} \leq 1 + y(t) + y(t-t^*) + \ldots + y(t-(\lceil \frac{q^*}{t*} \rceil - 1)t^*) \qquad (27)$$

is valid for S^x *where* $y(t) = 1 - \sum_{u=1:\ell_{k,u}\geq\beta}^{p_k} x_{k,t-u+1}$.

Proof: Job i is said to be active in period t if $\sum_{u=1}^{\min\{p_i,q^*+h_i\}} x_{i,t-u+1} = 1$.
We suppose *w.l.o.g.* that jobs $1,2,\ldots,m \leq k-1$ are active at t, and that job $i+1$ starts before i for $i = m-1,\ldots,1$.
Let $T_0 = \{t, t-T^*, \ldots, t-(\lceil \frac{q^*}{t*} \rceil - 1)t^*\}$.
Thus the left hand side of the inequality takes the value of the number of active jobs m, and the right hand side is $1 + \sum_{\tau \in T_0} y(\tau)$.
We now make a series of simple observations:
Observation 1: As job $i+1$ requires α labour units for h_i periods and $2\alpha > L$, $s_i - s_i + 1 \geq h_i + 1$.
Observation 2: As job m is active, $s_m \geq t - \min\{p_m, q^* + h_i\} + 1$. Thus, $s_m \geq t - q^* - h_m + 1$.

Observation 3: From observations 1 and 2, $s_{m-1} \geq s_m + h_m \geq t - q^* + 1$.
We now associate the interval $T_i = \{s_i, s_i + 1, \ldots, \min[t, s_i + t^* - 1]\}$ with each job. $i = 1, \ldots, m-1$.
Observation 4: All the jobs $m, m-1, \ldots, i+1$ started before job i, are still active in period t, and thus they are active during the interval T_i.
Observation 5: The intervals T_i are disjoint because the last period of T_{i+1} is $s_{i+1} + t^* - 1$, and the first of T_i is s_i. From Observation 1, $s_{i+1} + t^* - 1 \leq s_i - h_{i+1} + t^* - 1 \leq s_i - 1 < s_i$.
Observation 6: During interval T_i, the labour available for job $k \leq L - \alpha - \xi < \beta$ as job m is active throughout T_i.
Observation 7: The last element $s_{m-1} + t^* - 1$ of T_{m-1} satisfies

$$\begin{aligned} s_{m-1} + t^* - 1 &\geq t - q^* + 1 + t^* - 1 \quad \text{(by Observation 3)} \\ &= t + t^* - q^* \\ &= t - (\tfrac{q^*}{t^*} - 1)t^* \\ &\geq t - (\lceil \tfrac{q^*}{t^*} \rceil - 1)t^* \end{aligned}$$

Claim: For $i = 1, \ldots, m-1$, $\exists$ distinct integers $f_i \in \{0, \ldots, \lceil \frac{q^*}{t^*} \rceil - 1\}$ such that $y(t - f_i t^*) = 1$.

Proof of the claim: We have shown that the disjoint intervals $T_{m-1}, T_{m-2}, \ldots, T_1$ are of length t^*, except possibly for T_1.
The periods in T_0 are also equally spaced at intervals t^* between $t - (\frac{q^*}{t^*} - 1)t^*$ and t^*. T_{n-1} either contains $t - (\frac{q^*}{t^*} - 1)t^*$ or lies in $[t - \lceil \frac{q^*}{t^*} \rceil - 1)t^* + 1, t]$.
$T_{m-2}, \ldots, T_2$ lie in $[t - (\lceil \frac{q^*}{t^*} \rceil - 1)T^*, t]$.
T_1 either contains t, or is of length t^* and lies in $[t - \lceil \frac{q^*}{t^*} \rceil - 1)t^*, t]$.
Thus, there exists f_i such that $t - f_i t^* \in T_i$.
Now, by Observation 6, less than β units of labour are available in $t - f_i t^*$ and thus $y(t - f_i t^*) = 1$.
Finally, the right-hand side

$$1 + \sum_{r=0}^{(\lceil \frac{q^*}{t^*} \rceil - 1)} y(t - r t^*) \geq 1 + \sum_{i=1}^{m-1)} y(t - f_i t^*) \geq m,$$

and the inequality is valid. □

3.2 Surrogate Inequalities

To find inequalities maximising the number of jobs started in the first τ periods, it suffices to solve

$$\gamma_\tau = \max\{\sum_{j \in N} \sum_{t \leq \tau} x_{jt} : (x, s) \in S^x\}.$$

An upper bound $\bar{\gamma}_\tau$ can be obtained by partially dropping the integrality constraints on the variables x_{jt}, for example for $t > \tau$. The result is a valid inequality

$$\sum_{j \in N} \sum_{t \leq \tau} x_{jt} \leq \bar{\gamma}_\tau.$$

For values of $\tau \leq 20$, $\bar{\gamma}_\tau$ is obtained rapidly, and the resulting inequality has some effect in tightening the formulation.
A similar idea is to examine the maximum amount of labour used, or the number of jobs active in any interval of μ periods.

4 Computational Results with a Standard MIP System

As indicated in the Introduction, it has been observed that for tightly constrained instances where the makespan exceeds the longest path bound ξ_{PB},

1. typically $\xi_{LP} = \xi_{PB}$ for all the formulations weak or strong presented in Section 2
2. using a commercial MIP system with standard dichotomous branching on variables x_{it}, X_{it}, z_{it} or Z_{it}, the best lower bound hardly moves during thousands or tens of thousands of nodes.
3. MIP fails to find feasible solutions unless the horizon T is significantly greater than the makespan.
4. at least three heuristic approaches have been developed
   (a) constraint logic programming [7]
   (b) tabu search [2]
   (c) Order Heuristics based on i) using the LP solution to obtain an ordering of the jobs ii) use the ordering to construct a feasible solution iii) iterate using an exchange heuristic on the ordering iv) insert into a specialised Branch and Bound routine in which the heuristic is called at each node and specialised branching choices are implemented [10].

   All three heuristics appear to perform well in that each of them finds an optimal solution of instance b24 presented below.
5. Earlier work with an integer programing approach has shown that the knapsack inequalities of Proposition 10 can be added easily to the model, but have little immediate effect on the lower bounds. For instance b24, the inequalities of Propositions 7 and 11 are applicable. Combined with the block formulation and strong precedence constraints, the initial lower bound is raised from 59 to 65, and optimality has been proven for this instance [13].

Below we present our recent results obtained with an MIP system (XPRESS) running on a PC (Pentium, 166 MHz) workstation. First we attempt to tackle the problem directly.

4.1 Direct MIP Approach

The direct approach is to solve the MIP for a certain time of number of nodes, and then examine the quality of the best lower and upper bounds obtained. In Tables 1 and 2 we show results obtained using the weak and strong precedence formulations. For each instance PB denotes the longest path lower bound, T the time horizon chosen for the instance, m, n the number of rows and columns, LP the linear programming bound, LB and UB the lower and upper bounds on termination, and *secs* the time spent in branch and bound. For the weak formulation (Table 1), the number of nodes was set to 10000, and for the strong formulation (Table 2) to 5000. The formulations are the basic x formulation with just the a priori addition of simple knapsack cuts from Proposition 10 with $r = 1$, except for instance $b21m$ where some inequalities with $r = 2$ have also been added.

Instance	PB	T	m	n	LP	LB	UB	secs
b21	39	50	221	410	39.4	44.0	-	1104
b24	59	70	211	497	59.0	59.1	-	2029
b27	54	70	228	637	54.1	56.0	-	2522
b27m12	54	61	210	385	54.0	55.0	-	1389

Table 1: MIP Results: Weak Precedence Formulation- 10000 nodes

Instance	PB	T	m	n	LP	LB	UB	secs
b21	39	50	615	410	39.6	43.3	48	1834
b24	59	70	624	497	59.0	59.5	-	1993
b27	54	70	878	637	54.2	55.2	-	13775*
b27m12	54	61	536	385	54.0	54.7	-	2498

$*$ stopped after 3000 nodes

Table 2: MIP Results: Strong Precedence Formulation - 5000 nodes

4.2 MIP Heuristics providing Lower and Upper Bounds

Here we describe our efforts to develop MIP based heuristics that provide nontrivial lower and upper bounds in a reasonable time. Throughout we use the basic job x-formulation. On certain instances, improvements can be obtained by switching to the block formulation.

The Modified Objective Heuristic

The heuristic given below is based on the x-formulation with weak precedence constraints (5) as described in Section 2.1. The objective function is modified so that ξ equals to $\max \sum_t t^2 \, x_{jt}$, or $\max$ / $\min \sum_j s_j = \sum_t t \, x_{jt}$.

Heuristic:

Step 0: Choose an initial time horizon T.

Step 1: Solve the resulting **IP**.

Step 2: If **IP** is infeasible increase T to $T+1$, and return to **Step 1**.
Otherwise stop the solution of **IP** when a feasible solution is found. If $\overline{\xi}$ is the makespan of this feasible solution, decrease T below $\overline{\xi}$, and return to 3 **Step 1**.

As motivation for this heuristic, we have

(i) the LP is solved much quicker with the weak formulation, so the LP should only be tightened with strong valid inequalities that significantly affect the bounds or the running time

(ii) the LP solves much quicker with the modified objective than with the makespan objective.

One possible reason for (ii) is that the LP with the makespan objective is highly degenerate, and the objective function is flat. The LP bound does not change from one node to the next, and as a result no intelligent branching choices are made. With the modified objective, the LP bound decreases from node to node, and so branching decision remain coherent from one branch to the next.
The behaviour of the heuristic on 4 instances is shown in Table 3.

The Relax and Fix heuristic

Consider a partition $(T_1, T_2, \ldots, T_r)$ of the interval $[1, T]$, or a partition $(N_1, N_2, \ldots, N_r)$ of the job set N. Let $(x^1, \ldots, x^r)$ be a corresponding partition of the variables x.

Relax and Fix heuristic:

Step i**:** Solve (MIP^i) given by

$$\xi^i = \min(st(n) : (x, st) \in P, x^j = x^j(i)\ j = 1, \ldots, i-1, x^i \in Z_+)$$

with optimal solution $x(i)$.
If $x(r)$ is a feasible solution of MIP^r, $\xi \leq \xi^r$. As MIP^1 is a relaxation, $\xi \geq \xi^1$.

Clearly the difficulty in solving each problem MIP^i depends on the size of the intervals T^i or N^i. In Table 4 we show the results obtained with $r = 3$ using equally spaced time intervals. Problem MIP^1 was run to optimality, while problems MIP^2 and MIP^3 were limited to 15 minutes in the branch-and-bound phase. The heuristic was programmed using EMOSL [5], a combined modelling and optimisation language, designed to facilitate the development of new algorithmic approaches. In the Table we give the lower bound LB provided by the first problem MIP^1, and the upper bound UB provided by the last problem MIP^3. T denotes the time horizon used for the instance. The weak formulation was used.
The test instances we have used are available in [3].

Instance	T	ξ	secs	nodes
b21	47	47	32	489
b21	46	Infeasible	320	6373
b24	73	73	63	188
b24	70	70	340	958
b24	62	Infeasible	534	1837
b24	60	Infeasible	4	17
b27	73	72	25	40
b27	71	70	25	44
b27	68	68	7520	10271
b27	62	Infeasible	738	961
b27m12	68	67	24	460
b27m12	66	66	21	476
b27m12	64	64	19	439
b27m12	61	61	142	4681
b27m12	59	59	6468	208734
b27m12	57	Infeasible	3338	112105
b27m12	56	Infeasible	221	9231

Table 3: Modified Objective Heuristic

Instance	LB	UB	T
b21	39.6	47	50
b24	61.0	73	75
b27	60.0	74	74
b27m12	55	60	64

Table 4: Relax and Fix Heuristic

5 Further Work

From the above results, the large number of variables and constraints in the LPs and the very long solution times for these LPs, especially with the makespan objective, it is very clear that to provide good lower bounds a dedicated branch-and-bound approach is necessary. Such a system would need to eliminate all superfluous variables (i.e. when a new feasible solution is found of value ξ, or branching occurs) all variables x_{jt} with $t > \xi$ should be eliminated, and the earliest and latest start times $e(j)$ and $f(j)$ updated. In addition branching on variables x_{jt} is clearly unsatisfactory. GUB branching is one possibility, but initial experiments were not convincing. The use of logical implications together with branching may be helpful, and higher priorities for branching on variables on the critical path may also be important. It is also clear that heuristic solutions should be incorporated to branch-and-bound procedures. LP orders like those described in [6] should be used as primal heuristic solutions during the execution of the algorithm. Moreover, these orders could also be used as initial solutions for tabu search heuristics ([2]) which could produce tighter upper bounds.
We are also implementing a modified tree enumeration procedure without bounding which will allow us to complete the modified objective heuristic in a single pass.

Acknowledgements

The authors are very grateful to Cristina Cavalcante for pointing out to some mistakes on earlier versions of this paper and also for the help in putting the text in its final form.

References

[1] M. Bartusch, R.H. Moehring and F.J. Rademacher, "Scheduling Project Networks with resource constraints and time windows",

[2] C. C. B. Cavalcante, C. C. de Souza, "A Tabu Search Approach for Scheduling Problem Under Labour Constraints". Technical Report IC-97-13, IC-UNICAMP, October 1997.

[3] C. C. B. Cavalcante. WWW page: http://www.dcc.unicamp.br/~cris/SPLC.html.

[4] E. Demeulemeester and W. Herroelen, A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, *Management Science* **38**, 1803-1818 (1992).

[5] Entity Modelling and Optimisation Subroutine Library, XPRESS-MP, Draft Reference Manual, Dash Associates, Leamington Spa, September 1997.

[6] M.X. Goemans, "Improved approximations algorithms for scheduling with release dates". 8^{th} ACM-SIAM Symposium on Discrete Algorithms, 1997.

[7] S. Heipcke, "Resource constrained job-shop scheduling with constraint nets - two case studies", Diploma thesis, Mathematisch Geographiische Fakultaet, Katholische Universitaet Eichstaett, January 1995.

[8] S. Heipcke and Y. Colambani, "A New Constraint Programming Approach to Large Scale Resource Constrained Scheduling", Workshop on Models and Algorithms for Planning and Scheduling Problems, Cambridge, UK, April 1997.

[9] M. Queyranne and A. S. Schulz, "Polyhedral approaches to machine scheduling", Preprint 408/1994, Department of Mathematic, Technical University of Berlin, Berlin, Germany, 1994.

[10] M. Savelsbergh, Y. Wang and L.A. Wolsey, "Computational experiments with a large-scale resource constrained project scheduling problem", Note, Georgia Institute of Technology, August 1996.

[11] Sousa J., Wolsey L.A.: Time-indexed formulations of non-preemptive single machine scheduling problems. *Mathematical Programming* **54** (1992) 353-367.

[12] van den Akker M.: LP-based solution methods for single-machine scheduling problems. Ph.D thesis, Technical University of Eindhoven, December 1994.

[13] Y. Wang, L.A. Wolsey, Scheduling with labour constraints, Note, CORE, Université Catholique de Louvain, May 1996.