A MILP model for single machine family scheduling with sequence-dependent batch setup and controllable processing times

Davide Giglio
Department of Informatics, Bioengineering, Robotics and Systems Engineering (DIBRIS)
University of Genova
Via Opera Pia 13, 16145 – Genova, Italy
davide.giglio@unige.it

Abstract

A mathematical programming model for a class of single machine family scheduling problems is described in this technical report, with the aim of comparing the performance in solving the scheduling problem by means of mathematical programming with the performance obtained when using optimal control strategies, that can be derived from the application of a dynamic programming-based methodology proposed by the Author. The scheduling problem is characterized by the presence of sequence-dependent batch setup and controllable processing times; moreover, the generalized due-date model is adopted in the problem. Three mixed-integer linear programming (MILP) models are proposed. The best one, from the performance point of view, is a model which makes use of two sets of binary variables: the former to define the relative position of jobs and the latter to define the exact sequence of jobs. In addition, one of the model exploits a stage-based state space representation which can be adopted to define the dynamics of the system.
1 Introduction

A mathematical programming model for the single machine family scheduling problem described in the paper “Optimal control strategies for single machine family scheduling with sequence-dependent batch setup and controllable processing times” [1] is here presented in order to compare the dynamic programming-based approach proposed in [1] with a different methodology. Among the possible formulations, a mixed-integer linear programming (MILP) model which makes use of two sets of binary variables (one to define the relative position of jobs and the other to define the exact sequence of jobs) is adopted; as a matter of fact, a preliminary performance analysis has shown that such a model outperforms other models which solve the same problem.

This technical report is organized as follows. The scheduling problem is described in section 2 whereas the MILP model is reported in section 3. The test of the model (on the numerical example considered in [1]) and an experimental analysis (on randomly-generated instances of different sizes of the problem) are in section 4 and 5, respectively. A comparison of the times required to find an optimal solution between the algorithm proposed in [1] and the other to define the exact sequence of jobs) is adopted; as a matter of fact, a preliminary performance analysis is assumed that if the setup time and the initial setup cost, respectively, when a job of class \( k \) is the first processed job. The set of values \( \pi = \{ \pi_k, \pi_k \} \) for any \( k \), \( k = 1, \ldots, K \), is specified. It is assumed that \( \pi_k \leq \pi_k + 1 \) for any \( k = 1, \ldots, K \) and \( i = 1, \ldots, N \), and that jobs are assigned to the due-dates according to the EDD (earliest due-date) rule, in accordance with the generalized due-date model.

The processing (or service) time of the \( i \)-th job of class \( P_k \), namely \( p_{k,i} = \min \{ p_{k,i}^{\text{low}}, p_{k,i}^{\text{nom}} \} \), \( k = 1, \ldots, K \), \( i = 1, \ldots, N_k \); it is assumed that, once the processing time of a job has been chosen, it cannot be changed during the service. \( p_{k,i}^{\text{nom}} \) is the nominal value of the processing time, which can be compressed up to \( p_{k,i}^{\text{low}} \). In accordance with a resource consumption model, the processing time can be expressed as \( p_{k,i}(u_{k,i}) = p_{k,i}^{\text{nom}} - \gamma_k u_{k,i} \), with \( 0 \leq u_{k,i} \leq \pi_k, \pi_k = (p_{k,i}^{\text{nom}} - p_{k,i}^{\text{low}})/\gamma_k \), where \( u_{k,i} \) is the amount of a continuous resource used to reduce the processing time of the \( i \)-th job of class \( P_k \), and \( \gamma_k \) is the positive compression rate for jobs of class \( P_k \).

A costly setup is required for the execution of two jobs of different classes. In this connection, let \( s_{k,h} \) (resp., \( s_{k,k} \)), \( k = 1, \ldots, K \), \( h = 1, \ldots, K \), be the setup time (cost) which has to be spent (paid) when switching from a job of class \( P_k \) to one of class \( P_h \), \( h \neq k \). No setup is required between the execution of two jobs of the same class (i.e., \( s_{k,k} = 0 \) and \( s_{k,h} = 0 \) \( h \neq k \)). Besides, the execution of the first job does not require any setup, regardless of the class. Then, \( s_{0,k} = 0 \) and \( s_{0,k} = 0 \), \( k = 1, \ldots, K \), having denoted by \( s_{0,k} \) and \( s_{0,k} \) the initial setup time and the initial setup cost, respectively, when a job of class \( P_k \) is the first processed job.

The performance criterion (cost function) to be minimized is the total weighted tardiness, the total weighted tardiness cost (of the continuous resource used to compress the processing time), and the total setup cost, namely

\[
\sum_{k=1}^{K} \sum_{i=1}^{N_k} \alpha_{k,i} T_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \beta_k \gamma_k u_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} s_{k,i} \pi \tag{1}
\]

where: \( \alpha_{k,i} \) is the unitary tardiness cost for the \( i \)-th job of class \( P_k \); \( T_{k,i} \) is the tardiness of the \( i \)-th job of class \( P_k \); \( \beta_k \) is the unitary cost related to the deviation from the nominal service time for the execution of jobs of class \( P_k \); \( \pi_{k,i} \) is the index of the class of the job which precedes the \( i \)-th job of class \( P_k \) in the job sequence, or is the value 0 if the \( i \)-th job of class \( P_k \) is the first processed job. The set of values \( \Pi = \{ \pi_{k,i}, k = 1, \ldots, K, i = 1, \ldots, N_k \} \) is sufficient to define a specific job sequence; moreover, let \( U = \{ u_{k,i}, k = 1, \ldots, K, i = 1, \ldots, N_k \} \). The total weighted tardiness depends on both \( \Pi \) and \( U \), as \( T_{k,i} = \max\{ S_{k,i}(\Pi, U) + s_{k,i} + p_{k,i}^{\text{nom}} - \gamma_k u_{k,i} - d_{k,i}, 0 \} \), being \( S_{k,i}(\Pi, U) \) the time instant at which the \( i \)-th job of class \( P_k \) starts its execution when values \( \pi_{k,i} \) and \( u_{k,i} \), \( k = 1, \ldots, K, i = 1, \ldots, N_k \) are adopted.

Davide Giglio
DIBRIS – University of Genova
The objective is to find a job sequence $\Pi$ and the values $u_{k,i}$, $k = 1, \ldots, K$, $i = 1, \ldots, N_k$, which minimize the cost function (1).

3 The mathematical programming model

3.1 Variables and parameters

3.1.1 Main decision variables

- $x_{h,j,k,i}$ : binary variable which defines the relative position of jobs
  - $x_{h,j,k,i} = 1$ if the $j$-th job of class $P_h$ is executed before the $i$-th job of class $P_k$
  - $x_{h,j,k,i} = 0$ otherwise
- $\delta_{h,j,k,i}$ : binary variable which defines the sequence of jobs
  - $\delta_{h,j,k,i} = 1$ if the $i$-th job of class $P_k$ is the next job after the execution of the $j$-th job of class $P_h$
  - $\delta_{h,j,k,i} = 0$ otherwise
- $u_{k,i}$ : amount of a continuous used to reduce the processing time of the $i$-th job of class $P_k$

3.1.2 Other decision variables

- $S_{k,i}$ : time instant at which the $i$-th job of class $P_k$ starts its execution
- $pt_{k,i}$ : processing time of the $i$-th job of class $P_k$
- $T_{k,i}$ : tardiness of the $i$-th job of class $P_k$
- $\Omega_{k,i}$ : setup cost which is paid for the $i$-th job of class $P_k$
- $\Lambda_{k,i}$ : setup time which is spent for the $i$-th job of class $P_k$

3.1.3 Parameters

- $N_k$ : number of jobs of class $P_k$
- $pt_{k,\text{nom}}$ : nominal value of the processing time of jobs of class $P_k$
- $pt_{k,\text{low}}$ : lowest value of the processing time of jobs of class $P_k$
- $dd_{k,i}$ : due-date for the $i$-th job of class $P_k$
- $st_{h,k}$ : setup time which is spent when switching from a job of class $P_h$ to a job of class $P_k$
- $sc_{h,k}$ : setup cost which is paid when switching from a job of class $P_h$ to a job of class $P_k$
- $\alpha_{k,i}$ : unitary tardiness cost for the $i$-th job of class $P_k$
- $\beta_{k}$ : unitary processing time’s deviation cost for jobs of class $P_k$
- $\gamma_{k}$ : processing time’s compression rate for jobs of class $P_k$
- $M$ : large and positive value (“big number” larger than an upper-bound of the maximum completion time)
3.2 The MILP model

\[ \text{3.2.1 Objective function} \]

\[
\min \sum_{k=1}^{K} \sum_{i=1}^{N_k} \alpha_{k,i} T_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \beta_{k} \gamma_k u_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \Omega_{k,i} \quad (2)
\]

\[ \text{3.2.2 Constraints} \]

\[
T_{k,i} \geq S_{h,i} + \Lambda_{k,i} + p t_{k,i} - d d_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (3)
\]

\[
\Omega_{k,i} = \sum_{h=1}^{K} \sum_{j=1}^{N_h} s c_{h,k} \delta_{h,j,k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (4)
\]

\[
\Lambda_{k,i} = \sum_{h=1}^{K} \sum_{j=1}^{N_h} s t_{h,k} \delta_{h,j,k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (5)
\]

\[
\sum_{h=1}^{K} \sum_{i=1}^{N_h} \sum_{i=1}^{N_k} \delta_{h,j,k,i} = \left( \sum_{k=1}^{K} \sum_{k=1}^{N_k} \right) - 1 \quad (6)
\]

\[
\sum_{h=1}^{K} \sum_{i=1}^{N_h} \delta_{h,j,k,i} \leq 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (7)
\]

\[
\sum_{k=1}^{K} \sum_{i=1}^{N_k} \delta_{h,j,k,i} \leq 1 \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad (8)
\]

\[
\sum_{j=1}^{i} \delta_{k,i,k,j} = 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (9)
\]

\[
\sum_{j=i+2}^{N_k} \delta_{k,i,k,j} = 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k - 2 \quad (10)
\]

\[
u_{k,i} \leq \frac{p t_{k,i}^\text{nom} - p t_{k,i}^\text{low}}{\gamma_k} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (11)
\]

\[
p t_{k,i} = p t_{k,i}^\text{nom} - \gamma_k u_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (12)
\]

\[
S_{h,i} \geq S_{h,j} + \Lambda_{h,j} + p t_{h,j} - M (1 - x_{h,j,k,i}) \quad (13)
\]

\[
S_{h,i} \geq S_{h,j} + \Lambda_{h,j} + p t_{h,j} - M x_{h,j,k,i} \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (h,j) \neq (k,i) \quad (14)
\]

\[
x_{k,j,k,i} = 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad \forall j = 1, \ldots, i - 1 \quad (15)
\]

\[
x_{k,j,k,i} = 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad \forall j = i, \ldots, N_k \quad (16)
\]

\[
x_{h,j,k,i} + x_{k,i,h,j} = 1 \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (h,j) \neq (k,i) \quad (17)
\]

\[
x_{h,j,k,i} + x_{k,i,l,m} + x_{l,m,h,j} \leq 2 \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (h,j) \neq (k,i) \quad (k,i) \neq (l,m) \quad (l,m) \neq (h,j) \quad (18)
\]
Consider the numerical example for the single machine scheduling problem proposed in [1], in which 4 jobs of class $P_1$ and 3 jobs of class $P_2$ must be executed. The data which characterize the problem are the following.

The proposed problem is a mixed-integer linear mathematical programming problem (MILP). The number of binary variables is $2(\sum_{k=1}^K N_k)^2$. 

### 4 Test of the model

The proposed model is tested on a set of randomly generated instances. The results are compared with those obtained by other algorithms. The proposed model performs better in terms of solution quality and computational time. The model is also tested on real-world instances from the literature, and it outperforms other models in terms of solution quality and computational time.
A MILP model for single machine family scheduling
with sequence-dependent batch setup and controllable processing times

- Due-dates and unitary tardiness costs

| class $P_1$ | class $P_2$ |
|-------------|-------------|
| $\alpha_{1,1} = 0.75$ $dd_{1,1} = 19$ | $\alpha_{2,1} = 2$ $dd_{2,1} = 21$ |
| $\alpha_{1,2} = 0.5$ $dd_{1,2} = 24$ | $\alpha_{2,2} = 1$ $dd_{2,2} = 24$ |
| $\alpha_{1,3} = 1.5$ $dd_{1,3} = 29$ | $\alpha_{2,3} = 1$ $dd_{2,3} = 38$ |
| $\alpha_{1,4} = 0.5$ $dd_{1,4} = 41$ | — | — |

- Processing time bounds, unitary costs related to the deviation from the nominal processing times, and compression rates

| class $P_1$ | class $P_2$ |
|-------------|-------------|
| $\beta_1 = 1$ $pt_{1}^{low} = 4$ $\gamma_1 = 1$ | $\beta_2 = 1.5$ $pt_{2}^{low} = 4$ $\gamma_2 = 1$ |
| $pt_{1}^{nom} = 8$ | $pt_{2}^{nom} = 6$ |

- Setup times and costs

| times | costs |
|-------|-------|
| $st_{1,1} = 0$ $st_{1,2} = 1$ | $sc_{1,1} = 0$ $sc_{1,2} = 0.5$ |
| $st_{2,1} = 0.5$ $st_{2,2} = 0$ | $sc_{2,1} = 1$ $sc_{2,2} = 0$ |

By solving such an instance of the MILP problem, the following optimal solution is found.

| $x_{h,j,k,i}$ | $(k,i)$ | $(k,i)$ |
|---------------|---------|---------|
| $x_{1,1,k,i}$ | 1,1 | 1,2 | 1,3 | 1,4 | 2,1 | 2,2 | 2,3 |
| 1,1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1,2 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1,3 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1,4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| $\delta_{h,j,k,i}$ | $(k,i)$ | $(k,i)$ | job $(k,i)$ | $u_{k,i}$ |
|---------------------|---------|---------|-------------|---------|
| $\delta_{1,1,k,i}$ | 1,1 | 1,2 | 1,3 | 1,4 | 2,1 | 2,2 | 2,3 | 1,1 | 4 |
| 1,2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1,2 | 0.5 |
| 1,3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1,3 | 3 |
| 1,4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1,4 | 0 |
| 2,1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2,1 | 0 |
| 2,2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2,2 | 0 |
| 2,3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2,3 | 0 |
With such values of the main decision variables, the optimal values of the other decision variables are

| job \((k, i)\) | \(S_{k, i}\) | \(p_{k, i}\) | \(T_{k, i}\) | \(\Omega_{k, i}\) | \(\Lambda_{k, i}\) |
|------------|---------|--------|--------|--------|--------|
| 1, 1       | 12      | 4      | 0      | 1      | 0.5    |
| 1, 2       | 16.5    | 7.5    | 0      | 0      | 0      |
| 1, 3       | 24      | 5      | 0      | 0      | 0      |
| 1, 4       | 36      | 8      | 3.5    | 1      | 0.5    |
| 2, 1       | 0       | 6      | 0      | 0      | 0      |
| 2, 2       | 6       | 6      | 0      | 0      | 0      |
| 2, 3       | 29      | 6      | 0      | 0.5    | 1      |

The optimal job sequence provided by the solution is

\[ J_{2,1} \rightarrow J_{2,2} \rightarrow J_{1,1} \rightarrow J_{1,2} \rightarrow J_{1,3} \rightarrow J_{2,3} \rightarrow J_{1,4} \]

which is the same obtained with the methodology proposed in the paper in the nominal case. Moreover, the total cost of the optimal solution is 11.75, which corresponds to the value of the optimal cost-to-go in the initial state at \(t_0 = 0\), namely \(J_{0,0}(0)\), provided in [1]. However, it is worth noting that the optimal values of the decision variables (and thus the optimal job sequence) represent an open-loop solution that, when applied, maintains its validity as long as no disruption affects the nominal system behavior. Instead, the solution provided in [1] is a closed-loop solution that is able to give the optimal decisions for any actual machine behavior.

5 Experimental analysis

An experimental analysis has been carried out to test the performance of the MILP model in solving the considered family scheduling problem. Randomly-generated instances of different sizes have been solved. All the parameters but \(\gamma_k\) (which has been assumed deterministic with \(\gamma_k = 1 \forall k = 1, \ldots, K\)) have been considered stochastic variables uniformly distributed in the interval \([a, b]\), with \(a\) and \(b\) reported in the following table.

| parameter \(p_{k, i}^{\text{nom}}\) | Uniform distribution \([a, b]\) | note |
|-------------------------------|-------------------------------|------|
| \(p_{k, i}^{\text{low}}\)     | \(6\)                         | 10   |
| \(dd_{k, i}\)                | \(dd_{k, i-1} + 0.5\)         | \(dd_{k,0} = 10\) |
| \(st_{k,k}\)                | 1                             | 3    |
| \(sc_{k,k}\)                | 0.5                           | 2.5  |
| \(\alpha_{k, i}\)           | 0.5                           | 2.5  |
| \(\beta_k\)                 | 0.5                           | 2.5  |

Only small instances of the problem have been taken into account, which can be solved in seconds or minutes. The results of such an analysis follow.

2 classes – 5 jobs per class (Number of problems solved: 100)

| Problem size | Computational time (seconds) |
|--------------|-------------------------------|
| Binary variables | 200 | min | 0.14 |
| Other variables   | 61   | max | 0.86 |
| Constraints       | 1227 | mean | 0.29 |
|                  |   | stdev | 0.09 |
A MILP model for single machine family scheduling
with sequence-dependent batch setup and controllable processing times

2 classes – 10 jobs per class  (Number of problems solved: 100)

| Problem size | Computational time (seconds) |
|--------------|-----------------------------|
| Binary variables | 800                      | min | 8.17 |
| Other variables   | 121                       | max | 51.39 |
| Constraints       | 8757                      | mean| 18.91 |
|                  |                            | stdev| 7.69 |

2 classes – 15 jobs per class  (Number of problems solved: 20)

| Problem size | Computational time (seconds) |
|--------------|-----------------------------|
| Binary variables | 1800                      | min | 181.39 |
| Other variables   | 181                       | max | 5013.48 |
| Constraints       | 28587                      | mean| 1466.39 |
|                  |                            | stdev| 1204.22 |

3 classes – 5 jobs per class  (Number of problems solved: 60)

| Problem size | Computational time (seconds) |
|--------------|-----------------------------|
| Binary variables | 450                      | min | 16.36 |
| Other variables   | 91                       | max | 232.77 |
| Constraints       | 3790                      | mean| 42.63 |
|                  |                            | stdev| 29.03 |

Bigger instances of the scheduling problem take several hours to be solved.

6 Comparison with the dynamic programming-based approach

The performance of the dynamic programming-based (DP) approach proposed in [1] and the performance of the MILP approach have been compared on some specific instances of the scheduling problem.

- DP approach → Implemented and solved with Matlab R2014a
- MILP approach → Implemented and solved with Cplex 12.6 (with standard settings)

The results of such a comparison follow.

2 classes – 5 jobs per class

| Time to find an optimal solution (seconds) | DP Problem size | MILP Problem size |
|------------------------------------------|-----------------|-------------------|
| DP approach                             | 0.52            | State space nodes | 61 |
| MILP approach                           | 0.25            | Other variables   | 61 |
|                                         |                 | Constraints       | 1227 |
|                                         |                 | MILP nodes        | 527 |

Davide Giglio  DIBRIS – University of Genova
A MILP model for single machine family scheduling
with sequence-dependent batch setup and controllable processing times

| 2 classes – 10 jobs per class | 2 classes – 15 jobs per class | 2 classes – 20 jobs per class | 3 classes – 5 jobs per class | 3 classes – 10 jobs per class | 4 classes – 5 jobs per class |
|--------------------------------|--------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
| **Time to find an optimal solution (seconds)** | **DP Problem size** | **MILP Problem size** | **DP Problem size** | **MILP Problem size** | **DP Problem size** |
| DP approach | 1.47 | State space nodes | 221 | Binary variables | 800 | Other variables | 121 | Constraints | 8757 | MILP nodes | 13580 |
| MILP approach | 11.06 |  |  |  |  |  |  |  |  |  |
| **Time to find an optimal solution (seconds)** | **State space nodes** | **Time to find an optimal solution (seconds)** | **State space nodes** | **Time to find an optimal solution (seconds)** | **State space nodes** |
| DP approach | 3.17 | 481 | 5.87 | 841 | 4.99 | 541 | n.a. (> 50000) | 182.84 | 541 | n.a. (> 50000) |
| MILP approach | 2903.38 |  | n.a. (> 50000) |  |  |  |  |  |  |  |
| **Time to find an optimal solution (seconds)** | **State space nodes** | **Time to find an optimal solution (seconds)** | **State space nodes** | **Time to find an optimal solution (seconds)** | **State space nodes** |
| DP approach | 54.69 | 3631 | 91.31 | 4321 | 91.31 | 4321 | n.a. (> 50000) |  |  |  |
| MILP approach | n.a. (> 50000) |  | n.a. (> 50000) |  | n.a. (> 50000) |  |  |  |  |  |
The DP approach outperforms the MILP approach. Only in the case of very small instances of the problem, such as 2 classes and \( \leq 5 \) jobs, the performance of the two approaches is comparable (both approaches solve the problem in less than 1 second), but in all the other cases the DP approach has significantly better performance than the MILP approach.

7 Other mathematical programming models

Two different mixed-integer linear programming models are proposed in this section. The former is similar to the one proposed in subsection 3.2 but it uses only the set of binary variables \( \delta_{h,j,k,i} \); the latter exploits the stage-based state space representation, as introduced in [1]. However, these models do not show better performance (in terms of computational times to solve the considered instances of the scheduling problem) than that obtained with the model in subsection 3.2 of this technical report.

7.1 MILP model n.2

7.1.1 Variables and parameters

Let, in addition to the variables and parameters introduced in subsection 3.1:

- \( C_{k,i} \) : completion time of the \( i \)-th job of class \( P_k \) (other decision variable)

7.1.2 The model

An alternative MILP model is the following.

\[
\begin{align*}
\min & \quad \sum_{k=1}^{K} \sum_{i=1}^{N_k} \alpha_{k,i} T_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \beta_k \gamma_k u_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \Omega_{k,i} \\
\text{subject to} & \\
C_{k,i} & = S_{k,i} + \Lambda_{k,i} + p_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
T_{k,i} & \geq C_{k,i} - dd_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
\Omega_{k,i} & = \sum_{h=1}^{K} \sum_{j=1}^{N_h} sc_{h,k} \delta_{h,j,k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
\Lambda_{k,i} & = \sum_{h=1}^{K} \sum_{j=1}^{N_h} st_{h,k} \delta_{h,j,k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
\sum_{h=1}^{K} \sum_{i=1}^{N_k} \sum_{j=1}^{N_h} \delta_{h,j,k,i} & = \left( \sum_{k=1}^{K} N_k \right) - 1 \\
\sum_{h=1}^{K} \sum_{j=1}^{N_h} \delta_{h,j,k,i} & \leq 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
\sum_{k=1}^{K} \sum_{i=1}^{N_k} \delta_{h,j,k,i} & \leq 1 \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \\
\sum_{j=1}^{N_j} \delta_{k,i,k,j} & = 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k
\end{align*}
\]
The proposed problem is a mixed-integer linear mathematical programming problem (MILP). The number of binary variables is 7. The MILP model n.3 is stated as:

\[ \sum_{j=i+1}^{N_k} \delta_{k,i,k,j} = 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k - 2 \quad (37) \]

\[ u_{k,i} \leq \frac{pt_{k,\text{nom}} - pt_{k,\text{low}}}{\gamma_k} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (38) \]

\[ pt_{k,i} = pt_{k,\text{nom}} - \gamma_k u_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (39) \]

\[ S_{k,i} \geq C_{h,j} - M (1 - \delta_{h,j,k,i}) \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (h,j) \neq (k,i) \quad (40) \]

\[ C_{k,i} \geq M \sum_{h=1}^{K} \sum_{j=1}^{N_h} \delta_{h,j,k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (41) \]

\[ \delta_{h,j,k,i} \in \{0, 1\} \quad \forall h = 1, \ldots, K \quad \forall j = 1, \ldots, N_h \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (42) \]

\[ u_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (43) \]

\[ S_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (44) \]

\[ C_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (45) \]

\[ pt_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (46) \]

\[ T_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (47) \]

\[ \Omega_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (48) \]

\[ \Lambda_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad (49) \]

Constraint (29) computes the completion time of the \( i \)-th job of class \( P_k \) which is executed. Constraints (30) - (39) have the same meanings of constraints (3) - (12) described in subsection 3.2. Constraint (40) ensures that the \( i \)-th job of class \( P_k \) is scheduled after the \( j \)-th job of class \( P_h \) when \( \delta_{h,j,k,i} = 1 \). Constraint (41) determines the completion time of the first job which is executed (since the first job is the only job for which \( \delta_{h,j,k,i} = 0 \)). Finally, constraints (42) - (49) define the type of the decision variables.

The proposed problem is a mixed-integer linear mathematical programming problem (MILP). The number of binary variables is \((\sum_{k=1}^{K} N_k)^2\).

### 7.2 MILP model n.3

The scheduling model can be represented through a state space model in which the system state, when a new decision has to be taken, namely at time instant \( t_j, j = 0, 1, \ldots, N - 1 \), with \( N = \sum_{k=1}^{K} N_k \), is the \((N+2)\)-dimensional vector \( \mathbf{x}_j = [n_{1,j}, \ldots, n_{K,j}, h_j, t_j]^{T} \), being \( n_{k,j}, k = 1, \ldots, K \), the number of jobs of class \( P_k \) already completed at time instant \( t_j \), and \( h_j \) the class of the last served job. As a matter of fact, decision instants are discrete in time; they correspond to the initial time instant and the instants at which a job is completed, but for the last one.

It is apparent that the system state does not change between two successive decision instants. The \((N+1)\)-tuple \((n_{1,j}, \ldots, n_{K,j}, h_j)\) will be referred to as the discrete part of the system state vector; it can be represented through a stage-based state transition graph in which the states at the \( j \)-th stage, \( j \in \{0, 1, \ldots, N - 1\} \), are those such that \( \sum_{k=1}^{K} n_{k,j} = j \). An example of a state transition graph is reported in Fig. 1 for the case of two classes of jobs, with \( N_1 = 4 \) and \( N_2 = 3 \).

The system state evolves with the application of the control \( u_{j} = f(\mathbf{x}_j) \) at the decision instants \( t_j, j = 0, 1, \ldots, N - 1 \). The control, which is a function of the system state, is relevant to the choice of the class of the next job to be
A MILP model for single machine family scheduling with sequence-dependent batch setup and controllable processing times

Figure 1: State transition graph of the discrete part of the system state (in the case $K = 2$, with $N_1 = 4$, and $N_2 = 3$)

served and the value of the processing time. Then, $w_j = [\delta_{1,j}, \ldots, \delta_{K,j}, \tau_j]^T$, where $\delta_{k,j} \in \{0, 1\}$, $k = 1, \ldots, K$, is a binary decision variable whose value is 1 if a job of class $P_k$ is selected for the $(j+1)$-th service, and 0 otherwise, and $\tau_j$ is the processing time for the selected job. Obviously, it must be $\sum_{k=1}^{K} \delta_{k,j} = 1 \forall j = 0, 1, \ldots, N - 1$. Moreover, it is worth noting that the choice of considering the processing time $\tau_j$ as a decision variable instead of the amount of the continuous resource used to compress the processing time, namely $u_j$, does not modify the matter of the problem, as $u_j = \frac{1}{\sum_{k=1}^{K} \gamma_k \delta_{k,j}} (\sum_{k=1}^{K} p_{k,\text{nom}} \delta_{k,j} - \tau_j)$.

The state equations for the considered model are

$$
\mathbf{x}_{j+1} = \begin{bmatrix} n_{1,j+1} \\ \vdots \\ n_{K,j+1} \\ h_{j+1} \end{bmatrix} = \begin{bmatrix} n_{1,j} + \delta_{1,j} \\ \vdots \\ n_{K,j} + \delta_{K,j} \\ \sum_{k=1}^{K} k \delta_{k,j} \end{bmatrix}
$$

(50)

and the initial state vector is $\mathbf{x}_0 = [0, \ldots, 0, 0]^T$.

The objective is to find values $\delta_{k,j}$, $k = 1, \ldots, K$, and $\tau_j$, $j = 0, \ldots, N - 1$, that minimize the objective function

$$
\sum_{j=0}^{N-1} \left[ \alpha_{h_{j+1},n_{h_{j+1},j+1}} \max \{ t_{j+1} - d d_{h_{j+1},n_{h_{j+1},j+1}}, 0 \} + \beta_{h_{j+1}} (p_{h_{j+1},\text{nom}} - \tau_j) + s c_{h_{j+1},h_{j+1}} \right]
$$

(51)

### 7.2.1 Variables and parameters

Let, in addition to the variables and parameters introduced in section 3.1.
A MILP model for single machine family scheduling with sequence-dependent batch setup and controllable processing times

- \( x_{k,i,j} \): binary variable whose value is 1 if the \( i \)-th job of class \( P_k \) is executed at the \( j \)-th stage, and 0 otherwise (main decision variable). It is apparent that \( \delta_{k,j} = \sum_{i=1}^{N_k} x_{k,i,j} \) (then variable \( x_{k,i,j} \) can be employed in place of \( \delta_{k,j} \)).
- \( \tau_j \): processing time of the job which is executed at the \( j \)-th stage (main decision variable)
- \( \hat{\Omega}_j \): setup cost which is paid at the \( j \)-th stage (other decision variable)
- \( \hat{\Lambda}_j \): setup time which is spent at the \( j \)-th stage (other decision variable)
- \( \hat{S}_j \): time instant at which the job which is executed at the \( j \)-th stage starts its execution (other decision variable)
- \( \hat{C}_j \): completion time of the job which is executed at the \( j \)-th stage (other decision variable)
- \( N \): total number of jobs (parameter)

### 7.2.2 The model

An alternative MILP model is the following.

\[
\begin{align*}
\min & \quad \sum_{k=1}^{K} \sum_{i=1}^{N_k} \alpha_{k,i} T_{k,i} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \beta_k (p_{k,i}^{\text{nom}} - p_{k,i}) + \sum_{j=0}^{N-1} \hat{\Omega}_j \\
\text{subject to} & \quad T_{k,i} \geq C_{k,i} - dd_{k,i} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad p_{k,i} \geq p_{k,i}^{\text{low}} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad p_{k,i} \leq p_{k,i}^{\text{nom}} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad \hat{\Omega}_j \geq \hat{\Omega}_{h,k} \left( \sum_{i=1}^{N_h} x_{h,i,j-1} + \sum_{i=1}^{N_k} x_{k,i,j} - 1 \right) \quad \forall j = 1, \ldots, N - 1 \quad \forall h, k = 1, \ldots, K \\
& \quad \hat{\Omega}_0 = 0 \\
& \quad \hat{\Lambda}_j \geq \hat{\Lambda}_{h,k} \left( \sum_{i=1}^{N_h} x_{h,i,j-1} + \sum_{i=1}^{N_k} x_{k,i,j} - 1 \right) \quad \forall j = 1, \ldots, N - 1 \quad \forall h, k = 1, \ldots, K \\
& \quad \hat{\Lambda}_0 = 0 \\
& \quad \hat{S}_j = \hat{C}_{j-1} \quad \forall j = 1, \ldots, N - 1 \\
& \quad \hat{S}_0 = 0 \\
& \quad \hat{C}_j = \hat{S}_j + \hat{\Lambda}_j + \tau_j \quad \forall j = 0, \ldots, N - 1 \\
& \quad \tau_j \geq p_{k,i} - M (1 - x_{k,i,j}) \quad \forall j = 0, \ldots, N - 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad S_{k,i} \geq \hat{S}_j - M (1 - x_{k,i,j}) \quad \forall j = 0, \ldots, N - 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad C_{k,i} \geq \hat{C}_j - M (1 - x_{k,i,j}) \quad \forall j = 0, \ldots, N - 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \\
& \quad S_{k,i} \geq C_{k,i-1} \quad \forall k = 1, \ldots, K \quad \forall i = 2, \ldots, N_k 
\end{align*}
\]
A MILP model for single machine family scheduling
with sequence-dependent batch setup and controllable processing times

\[ \sum_{k=1}^{K} \sum_{i=1}^{N_k} x_{k,i,j} = 1 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (67)

\[ \sum_{j=0}^{N-1} \sum_{i=1}^{N_k} x_{k,i,j} = N_k \quad \forall k = 1, \ldots, K \]  \hspace{1cm} (68)

\[ \sum_{j=0}^{N-1} x_{k,i,j} = 1 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \]  \hspace{1cm} (69)

\[ x_{k,i,j} \in \{0, 1\} \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (70)

\[ \tau_j \geq 0 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (71)

\[ S_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \]  \hspace{1cm} (72)

\[ C_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \]  \hspace{1cm} (73)

\[ pt_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \]  \hspace{1cm} (74)

\[ T_{k,i} \geq 0 \quad \forall k = 1, \ldots, K \quad \forall i = 1, \ldots, N_k \]  \hspace{1cm} (75)

\[ \tilde{\Omega}_j \geq 0 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (76)

\[ \tilde{A}_j \geq 0 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (77)

\[ \tilde{S}_j \geq 0 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (78)

\[ \tilde{C}_j \geq 0 \quad \forall j = 0, \ldots, N - 1 \]  \hspace{1cm} (79)

Constraint (53) computes, taking into account (75) and the presence of \( T_{k,i} \) in the cost function, the tardiness of the \( i \)-th job of class \( P_k \) which is executed. Constraints (54) and (55) set the range of allowed values of the processing time of the \( i \)-th job of class \( P_k \). Constraints (56) and (58) compute, respectively, the setup cost and the setup time which are paid/spent at the \( j \)-th stage. Constraints (57) and (59) set the initial setup cost and the initial setup time, respectively, to 0. Since it can be shown that an optimal solution of the considered single machine family scheduling problem is characterized by the absence of idle times between the execution of two subsequent jobs (even if the cost function is not of regular type), constraint (60) states that each job starts as soon as the previous one completes, and constraint (61) sets the start time of the first job to 0. Constraint (62) computes the completion time of the job which is executed at the \( j \)-th stage. Constraint (63) sets the processing time of the job which is executed at the \( j \)-th stage, taking into account the values of binary variables \( x_{k,i,j} \) as well as the range fixed by (54) and (55). Constraints (64) and (65) determine, respectively, the start time and the completion time of the \( i \)-th job of class \( P_k \) which is executed. Constraint (66) deals with the generalized due-date model as it ensures that, for what concerns class \( P_k \), the \( i \)-th job is scheduled after the \((i-1)\)-th job (that is, the \( i \)-th job starts after the completion time of the \((i-1)\)-th job). Constraint (67) guarantees that a job is executed only once. Constraint (68) ensures that all jobs are executed, whereas constraint (69) establishes that, in each stage, only one job can be served. Finally, constraints (70) and (79) define the type of the decision variables.

The proposed problem is a mixed-integer linear mathematical programming problem (MILP). The number of binary variables is \( \left( \sum_{k=1}^{K} N_k \right)^2 \).

References

[1] Davide Giglio. Optimal control strategies for single machine family scheduling with sequence-dependent batch setup and controllable processing times. Journal of Scheduling. Under review.