J Math Model Algor (2013) 12:117–133
DOI 10.1007/s10852-012-9196-2

Equipment Location in Machining Transfer Lines with Multi-spindle Heads

Olga Battaïa · Evgeny Gurevsky · Fatme Makssoud · Alexandre Dolgui

Received: 15 December 2010 / Accepted: 12 June 2012 / Published online: 12 August 2012

Abstract The considered problem appears when a machining line must be configured. It is necessary to define the number of workstations and the number of spindle heads at each workstation to be put in the line in order to produce a given part. This problem is known to be $\mathcal{NP}$-hard and, as a consequence, the solution time increases exponentially with the size of the problem. A number of pre-processing procedures are suggested in this article in order to decrease the initial problem size and thus shorten the solution time. A new algorithm for calculating a lower bound on the number of required equipment is also presented. A numerical example is given.

Keywords Pre-processing procedures · Equipment location · Lines configuration · Machining transfer lines

1 Introduction

The equipment location in production lines presents a complex combinatorial problem and effective mathematical tools are required to handle it [12–19]. In the considered paced serial machining lines, two types of equipment have to be located: workstations and spindle heads [1]. The goal is to find a configuration having the

O. Battaïa (✉) · E. Gurevsky · F. Makssoud · A. Dolgui
Laboratory for Informatics, Modeling and Optimization of Systems (LIMOS),
École Nationale Supérieure des Mines de Saint-Étienne, Henri Fayol Institute,
158, cours Fauriel, 42023 Saint-Étienne Cédex 2, France
e-mail: battaia@emse.fr

E. Gurevsky
e-mail: gurevsky@emse.fr

F. Makssoud
e-mail: makssoud@emse.fr

A. Dolgui
e-mail: dolgui@emse.fr

minimal equipment cost and such that a part launched in the beginning of the line can be completed by visiting sequentially all workstations of the line. The workstations are linked by an automatic handling device which is used for a synchronous parts transfer. The time span between two part movements is referred to as the *line cycle time* and is denoted T_0 [3].

At each workstation, a number of sequentially activated multi-spindle heads are installed. A multi-spindle head carries multiple tools for performing simultaneously a set of machining operations [9]. A set of operations assigned to the same spindle head is referred to as a *block of operations* (or *block*). To define the number of multi-spindle heads to be put at each workstation, the order of their activation and the set of operations to be performed by each one is a part of the equipment location problem.

The number of workstations is limited by the available plant area and the maximum affordable line investment cost. As a consequence, it is constrained by the maximum authorized value denoted by m_0. The number of spindle heads per workstation is restricted by design and construction constraints and cannot therefore exceed the maximum authorized value denoted by n_0.

The objective of the equipment location problem is to define the number of required workstations and the number of multi-spindle heads at each workstation taking into account existing technological constraints (detailed in Section 2) while minimizing the total investment cost. Various exact and approximate methods have been already proposed for solving this $\mathcal{NP}$-hard problem [10, 11]. The evaluation of their performances presented in [8] showed that exact methods are very sensible to the size of the problem which is defined by the number of operations to be assigned to spindle heads and workstations as well as by the number of existing constraints. A first attempt to decrease the size of the problem by aggregating some operations was made in [2]. In this article, new pre-processing procedures are suggested as well as an improved algorithm for calculating a lower bound on the number of required equipment is developed. A numerical example is given in Section 4. Conclusion remarks are presented in Section 5.

2 Problem Statement

2.1 Cycle Time Constraint

The machining process is defined by a set $\mathbf{N}$ of operations required to produce a given part. Each operation deals with the removal of material and thus needs a cutting tool which is mounted on a spindle head.

Each operation $j \in \mathbf{N}$ is characterized by two parameters: required working stroke length λ_j and recommended feed per minute f_j. For a spindle head performing a set of operations $N \subseteq \mathbf{N}$, different technological modes may be used [20]: either a single speed is applied to all tools [7] or machine attachments may be used to change the relative speed between the main spindle and the tools performing operations [11]. In the latter case, the tools of the same spindle head can machine at different speeds.

Let N_{kl} be the set of operations assigned to block l (spindle head) of workstation k ($N_{kl} \subseteq \mathbf{N}$). According to two technological modes described before, one of two following models for the block time calculation is used.

Model 1 If each operation is performed at its own speed (by use of machine attachments), then block time $t^b(N_{kl})$ is defined by the longest operation of N_{kl}:

$$t^b(N_{kl}) = \max\left\{\frac{\lambda_j}{f_j} : j \in N_{kl}\right\} + \tau^b,$$

where τ^b is a constant which gives an additional non-processing time required for the activation of a spindle head (it does not depend on set N_{kl}).

This model is also applicable for the case where operation times t_j, $j \in \mathbf{N}$, are given instead of machining parameters λ_j and f_j. Then the block time is calculated as follows:

$$t^b(N_{kl}) = \max\{t_j : j \in N_{kl}\} + \tau^b.$$

Model 2 The second model for the block time calculation is used if a single speed is applied for all operations performed by a spindle head. In this case, the common values $\Lambda(N_{kl})$ and $F(N_{kl})$ of working stroke length and feed per minute must be found for all operations grouped in a block. They are calculated as follows:

$$\Lambda(N_{kl}) = \max\{\lambda_j : j \in N_{kl}\}, \quad F(N_{kl}) = \min\{f_j : j \in N_{kl}\}.$$

Then, the block time is calculated as follows:

$$t^b(N_{kl}) = \frac{\Lambda(N_{kl})}{F(N_{kl})} + \tau^b.$$

Since all spindle heads of a workstation are applied sequentially to the part being machined, workstation time $t^w(N_k)$ is calculated as follows:

$$t^w(N_k) = \sum_{l=1}^{b_k} t^b(N_{kl}) + \tau^w,$$

where $N_k = \cup_{l=1}^{b_k} N_{kl}$ is the set of operations assigned to workstation k, b_k is the number of blocks at workstation k, and τ^w is a constant (the same for all workstations) which represents an additional non-processing time required for the part installation on a workstation.

The workstation time is subject to a productivity constraint and must not be greater than line cycle time T_0, i.e.

$$\max_{1 \le k \le m} t^w(N_k) \le T_0,$$

where m is the number of used workstations.

2.2 Operations Assignment Constraints

The incompatibilities among groups of operations, caused by the impossibility to mount the required tools on the same spindle head or to perform a given group of operations at the same workstation, prohibit the assignment of such groups of operations to a common block or workstation. These constraints are called the *block* and *workstation exclusion constraints*, respectively. They are represented by:

– Family E^b containing such sets of operations that cannot be grouped together in the same block,

– Family E^w composed of such sets of operations that must not be assigned together to the same workstation.

To avoid a loss of machining precision, some groups of operations must be executed without any part repositions, i.e. necessarily at the same workstation, but not imperatively in the same block. These constraints are called the *workstation inclusion constraints*. They are represented by family I^w consisting of sets of operations such that all operations of each set must be performed at the same workstation.

The elements of E^b and E^w are minimal sets (with respect to inclusion), i.e. if $e \in E^b$, then for any proper subset $e' \subset e$: $e' \notin E^b$ and all operations from e' can be grouped together into one block (similarly for E^w). On the other hand, the elements of I^w are maximal sets (with respect to inclusion), i.e. if $e \in I^w$, then for any set $e' \subset \mathbf{N}$ such that $e \subset e'$: $e' \notin I^w$.

Operations are subject to precedence constraints [5]. The particularity of precedence constraints in machining systems lies in the fact that if operation j is a successor of operation i, then two following cases are authorized: either operations i and j are performed simultaneously in a common block or the block containing operation j starts after the completion of the block containing operation i. In any case, operation j cannot start before the beginning of operation i.

Such a non-strict order relation can be represented by a directed acyclic graph $G = (\mathbf{N}, D)$. Arc $(i, j) \in \mathbf{N} \times \mathbf{N}$ belongs to set D if and only if operation j cannot precede operation i. The following sets can be determined for each operation $i \in \mathbf{N}$:

– Set $AP(i)$ of all its predecessors,
– Set $IP(i)$ of its immediate predecessors,
– Set $AS(i)$ of all its successors.

2.3 Solution Representation

A feasible solution of the considered problem can be represented by a collection $\{\{N_{11}, \ldots, N_{1b_1}\}, \ldots, \{N_{m1}, \ldots, N_{mb_m}\}\}$, where $m \leq m_0$ is the number of workstations; $b_k \leq n_0$ is the number of blocks for workstation k; N_{kl} $(k = 1, 2, \ldots, m;$ $l = 1, 2, \ldots, b_k)$ is the set of operations assigned to the l-th block of the k-th workstation.

An optimal solution is a feasible solution that minimizes the following function:

$$C_1 m + C_2 \sum_{k=1}^{m} b_k \rightarrow \min,$$

where constant C_1 represents the cost of installing a workstation which does not depend on the number of spindle heads located on it. This cost includes the investment cost for the workstation support, machine body, and space occupation cost. Each spindle head involves the cost represented by constant C_2. It includes the cost of basic spindle head elements: spindle motor, housing, base, column, spindle head mounting, and etc.

With all introduced notations, an instance of the considered problem is denoted as $\{\mathbf{N}, T_j^{op} (j \in \mathbf{N}), \tau^b, \tau^w, T_0, G, E^b, E^w, I^w, m_0, n_0, C_1, C_2\}$. Notation T_j^{op} is used to cover different types of data that can be known about operations. Even if in practice operations are usually given by λ_j and f_j ($j \in \mathbf{N}$), $T_j^{op} = \frac{\lambda_j}{f_j}$, they can be also characterized by their operation times $T_j^{op} = t_j$.

A MIP formulation of this $\mathcal{NP}$-hard problem has been introduced in [4]. Then, a number of exact and approximate methods have been proposed to solve it. State-of-the-art solution methods have been presented in [8, 10, 11]. The undertaken performance evaluations [8] showed that the required computational time to solve exactly the problem depends on the number of operations to be assigned, on the number of constraints, and on the used lower and upper bounds. A first attempt to decrease the number of operations by aggregating some of them was made in [2]. In the next section, new pre-processing procedures are developed. They have four objectives: firstly, to determine contradictory constraints, if they exist (Section 3.1); secondly, to reduce the number and the size of the given constraints (Section 3.2); thirdly, to calculate a lower bound on the total number of blocks and the intervals of blocks for each operation where it can be assigned (Section 3.3); and finally, to extract some operations (Section 3.4).

3 Pre-processing Methods

3.1 Input Data Compatibility

There may be problem instances containing contradictory constraints because of which no feasible solution exists for the corresponding optimization problems. This can be caused by a mistaken input data. The objective is to detect such cases before attempting to solve the problem. Four following conditions are suggested to do it, there is no feasible solution for a problem if at least one of them holds:

Condition 1 There exists $j \in \mathbf{N}$ such that $T_j^{op} > T_0 - \tau^w - \tau^b$.

The assignment of operation j even to a workstation with only one block violates the cycle time constraint. Simultaneous machining allows decrease the total execution time, but it is not possible to reduce the execution time of any single operation. As it was shown in Section 2, the block time is equal to the time of the longest operation included in it.

Condition 2 There exist $e_1 \in E^w$ and $e_2 \in I^w$ such that $e_1 \subseteq e_2$.

The inclusion constraints require performing all operations belonging to e_2 at the same workstation, while exclusion constraint e_1 forbids such an allocation due to $e_1 \subseteq e_2$. It is obvious that a solution respecting the both contradictory constraints cannot be found.

Condition 3 There exist $e_1 = \{i_1, j_1\} \in E^w$ and $e_2 = \{i_2, j_2\} \in I^w$ such that $i_1 \in AP(j_1)$ and the following conditions hold:

- $i_2 \in AP(i_1)$ or $i_1 = i_2$,
- $j_1 \in AP(j_2)$ or $j_1 = j_2$.

Exclusion constraint e_1 makes impossible to assign any predecessor of i_1 and any successor of j_1 to the same workstation. Therefore, a solution respecting contradictory constraints e_1 and e_2 does not exist.

Condition 4 There exist $e_1 = \{i, j\} \in E^b$ and $e_2 \in I^w$ such that $e_1 \subseteq e_2$ and $T_i^{op} + T_j^{op} > T_0 - \tau^w - 2\tau^b$.

Inclusion constraint e_2 requires performing operations i and j at the same workstation. Because of exclusion constraint e_1, they cannot be assigned to the same block. However, if they are assigned to two different blocks of the same workstation, then the cycle time constraint is violated. Therefore, a solution respecting all given constraints cannot be found.

3.2 Constraints Transformation

The workstation exclusion constraints can be considered stronger than the block exclusion constraints, since if two operations cannot be assigned to the same workstation, they obviously cannot be assigned to the same block. The following transformation of some block exclusion constraints to workstation exclusion constraints was suggested and proven in [6]: if there exists $e = \{i, j\} \in E^b$ such that $T_i^{op} + T_j^{op} > T_0 - \tau^w - 2\tau^b$, then e can be removed from E^b and added to E^w.

Here some new rules for the constraints transformation are suggested. The first two ones deal with the transformation of the workstation inclusion constraints.

Proposition 1 *If there exist $e_1, e_2 \in I^w$, $i, j \in e_1$, and $h \in e_2$ such that $i \in AP(h)$ and $h \in AP(j)$, then e_1 and e_2 can be united: $e_1 = e_1 \cup e_2$.*

Proof Because of precedence relations, operation h must be assigned only after operation i and before operation j. Inclusion constraint e_1 requires performing operations i and j at the same workstation. As a result, i, h, and j as well as all other operations from e_1 and e_2 have to be assigned to the same workstation in order to respect these inclusion and precedence constraints. □

Proposition 2 *If there exists $e \in I^w$ and $i, j, h \in e$ such that $i \in AP(j)$ and $j \in AP(h)$, then operation j can be deleted from e: $e = e \setminus \{j\}$.*

Proof Operation j can be deleted from e, since its assignment to the same workstation as i and h are provided by the precedence constraints. □

Propositions 1 and 2 aim in reducing the number and the size of the workstation inclusion constraints. Proposition 3 deals with the elimination of such inclusion constraints where the creation of macro-operations is possible.

Proposition 3 *If there exists $e = \{i, j\} \in I^w$ such that $T_i^{op} + T_j^{op} > T_0 - \tau^w - 2\tau^b$, then operations i and j have to be assigned to the same block and can be replaced by a macro-operation.*

Proof Inclusion constraint e requires performing operations i and j at the same workstation. Since $T_i^{op} + T_j^{op} > T_0 - \tau^w - 2\tau^b$, it is not possible to assign i and j to two different blocks without violating the cycle time constraint. Therefore, in order to respect inclusion constraint e and the cycle time constraint, operations i and j have to be assigned to the same block. This means that the assignment of operation $i(j)$

defines the assignment of operation $j(i)$ and, as a consequence, they can be replaced by a macro-operation. □

Finally, the exclusion constraints can be transformed by analyzing the precedence relations.

Proposition 4 *If there exist $e \in E^b(E^w)$ and $i, j, h \in e$ such that $i \in AP(j)$ and $j \in AP(h)$, then operation j can be deleted from e.*

Proof For respecting constraint e, it is sufficient that at least one operation from e is not assigned to the same block (workstation) as others. In any case, it cannot be operation j, since it is impossible to assign operations i and h to the same block (workstation) without assigning operation j to the same block (workstation) because of the precedence constraints. This allows removing operation j from e. □

Proposition 5 *If there exist $e_1 = \{i_1, j_1\}$ and $e_2 = \{i_2, j_2\}$ such that*

- $i_1 \in AP(j_1)$,
- $i_2 \in AP(i_1)$ *or* $i_1 = i_2$,
- $j_1 \in AP(j_2)$ *or* $j_1 = j_2$,

and one of the following cases holds:

1. $e_1, e_2 \in E^b$,
2. $e_1, e_2 \in E^w$,
3. $e_1 \in E^w$, $e_2 \in E^b$,

then e_2 can be removed either from E^b (Cases 1, 3) or from E^w (Case 2).

Proof Exclusion constraint e_1 makes impossible to assign any predecessor of i_1 and any successor of j_1 to the same block (workstation). Thus, all additional constraints concerning the impossibility of such assignments are impertinent and can be deleted. In the third case it is obvious that if operations i_1 and j_1 cannot be assigned to the same workstation, then any predecessor of i_1 and any successor of j_1 can be assigned neither to the same workstation nor to the same block. □

3.3 A Lower Bound Calculation

The first method for calculating lower bounds on the number of workstations and on the number of blocks has been proposed in [4]. In this paper, a new improved algorithm is presented to calculate more precisely a lower bound on the number of required blocks. To obtain this bound, the numbers of the earliest q_j^- and the latest q_j^+ blocks, where operation j can be assigned, are used. Since all existing blocks are applied sequentially to the part being machined, one can number the blocks in the order of their application to the part. It is supposed that there are n_0 blocks per workstation and, as a consequence, the first block of the first workstation has number 1, the first block of the second workstation number $n_0 + 1$, the first block of the k-th workstation number $(k - 1)n_0 + 1$ and so on until $m_0 n_0$ which is the upper bound on the number of blocks and the last number attributed to the last possible block of the last possible workstation.

The distance between operations i and j, denoted $d(i, j)$, is calculated in the following way:

$$d(i, j) = \begin{cases} 2, & \text{if } \{i, j\} \in E^w, \\ 1, & \text{if } \{i, j\} \in E^b, \\ 0 & \text{otherwise.} \end{cases}$$

The earliest block q_j^-, where operation j can be assigned, is found by Algorithm 1, where j_{cur} indicates if a new iteration must be carried out, $j_{\min}$ is a supplementary variable,

$$Q_{\max}(j) = \max\left\{q_i^- \oplus d(i, j) : i \in AP(j), \min\left\{q_i^-, q_h^-\right\} \oplus d(i, h) : i, h \in IP(j)\right\},$$

$$q \oplus d = \begin{cases} \left\lceil \dfrac{q}{n_0} \right\rceil n_0 + 1, & \text{if } d = 2, \\ q + d & \text{otherwise,} \end{cases}$$

and

$$W(q) = \left(\left\lceil \frac{q}{n_0} \right\rceil - 1\right) n_0 + 1$$

calculates the number of the first block of the workstation to which block q is assigned. Here $\lceil s \rceil$ is the smallest integer value greater than or equal to s.

Algorithm 1: Calculation of q_j^-, $j \in \mathbf{N}$

```
for j ← 1 to |N| do
    q_j^- ← 1
j_cur ← 1
repeat
    j_min ← j_cur, j_cur ← |N|
    for j ← j_min + 1 to |N| do
        q_j^- ← max{q_j^-, Q_max(j)}
    foreach e ∈ I^w do
        M(e) ← max{W(q_j^-) : j ∈ e}
        foreach j ∈ e do
            if q_j^- < M(e) then
                q_j^- ← M(e), j_cur ← min{j_cur, j}
until j_cur = |N|
```

To compute the latest block q_j^+, where operation j can be assigned, Algorithm 2 is used.

Algorithm 2: Calculation of q_j^+, $j \in \mathbf{N}$

```
Generate graph Ĝ by inverting the arcs of graph G
Calculate q̂_j^-, j ∈ N, basing on graph Ĝ
for j ← 1 to |N| do
    q_j^+ ← m_0 n_0 − q̂_j^- + 1
```

Then, obtained values can be refined by applying Algorithm 3. For this purpose, set $O(q)$ is at first calculated. Set $O(q)$ contains all operations that can be assigned to block q, i.e.

$$O(q) = \{ j : q \in [q_j^-, q_j^+] \}.$$

Algorithm 3 employs some additional notations which need to be introduced. Condition $(q_j^- - 1) \bmod n_0 \neq 0$ verifies if q_j^- is the first block of a workstation or not, where $a \bmod b$ is used to get the remainder of the division of an integer a by an integer b, q_{cur} indicates if a new iteration must be carried out, q^* keeps the initial value of q_j^-, AL contains studied operation j and the operations connected with it by the workstation inclusion constraints.

Algorithm 3: Improvement of q_j^- and q_j^+, $j \in \mathbf{N}$

```
j_cur ← 1, q_cur ← |N|
repeat
    j_min ← j_cur, j_cur ← |N|
    for j ← j_min + 1 to |N| do
        q* ← q_j^-
        if q_j^- ≥ q_cur and AP(j) ≠ ∅ then
            q_j^- ← max{q_j^-, Q_max(j)}, O(q_j^-) ← O(q_j^-) ∪ {j}
            if q* < q_j^- then
                for i ← q* to q_j^- − 1 do
                    O(i) ← O(i) \ {j}
        q* ← q_j^-
        if (q_j^- − 1) mod n_0 ≠ 0 then
            if Σ_{q=W(q_j^-)}^{q_j^- −1} min_{i∈O(q)} {T_i^op + τ^b} > T_0 − τ^w − τ^b − T_j^op then
                if there exists e ∈ I^w such that j ∈ e then
                    AL ← e
                else
                    AL ← {j}
                foreach i ∈ AL do
                    q_cur ← min{q_cur, q_i^-}, j_cur ← min{j_cur, i}, q_i^- ← W(q*) + n_0,
                    O(q_i^-) ← O(q_i^-) ∪ {i}
                for i ← q* to q_j^- − 1 do
                    O(i) ← O(i) \ AL
        while (q_j^+ − 1) mod n_0 ≠ 0 and
              Σ_{q=W(q_j^+)}^{q_j^+ −1} min_{i∈O(q)\{j}} {T_i^op + τ^b} > T_0 − τ^w − τ^b − T_j^op do
            O(q_j^+) ← O(q_j^+) \ {j}, q_j^+ ← q_j^+ − 1
until j_cur = |N|
```

Thus, a lower bound on the number of blocks and workstations is respectively calculated as:

$$\left| \left\{ q_j^- : q_j^+ \leq n_0 \left\lceil \frac{q_j^-}{n_0} \right\rceil, j \in \mathbf{N} \right\} \right|, \quad \left\lceil \frac{\max\{q_j^- : j \in \mathbf{N}\}}{n_0} \right\rceil.$$

The obtained sets $O(q)$ can be also used to verify the pertinence of certain constraints. Let N^*_{kl} contain all operations that can be assigned to block l of workstation k. Such sets can be easily obtained from sets $O(q)$: $N^*_{kl} = O(q)$, where $k = \lceil \frac{q}{n_0} \rceil$,

$$l = \begin{cases} n_0, & \text{if } q \bmod n_0 = 0, \\ q \bmod n_0 & \text{otherwise.} \end{cases}$$

Proposition 6 *If the first model for the block time calculation is used (see Section 2.1) and for workstation k:*

$$\sum_{l=1}^{n_0} \left(\max_{i \in N^*_{kl}} T_i^{op} + \tau^b \right) \leq T_0 - \tau^w \tag{1}$$

or if the second model is used and for workstation k:

$$\sum_{l=1}^{n_0} \left(\max_{i,j \in N^*_{kl}} \frac{\lambda_i}{f_j} + \tau^b \right) \leq T_0 - \tau^w, \tag{2}$$

then the cycle time constraint for workstation k is not pertinent.

Proof It is easy to see that the left-hand side of Eqs. 1 or 2 is an upper bound of the execution time on the k-th workstation. This means that real workstation time $t^w(N_k)$ can exceed neither this value nor, as a consequence, line cycle time T_0. Therefore, for workstation k the cycle time constraint is impertinent and can be removed from the model. □

Note If Eqs. 1 or 2 holds for all $k = 1, \ldots, m_0$, then either the cycle time constraint must be removed from the problem's model or T_0 can be decreased.

Proposition 7 *If the first model for the block time calculation is used and for workstation k:*

$$\sum_{l=1}^{n_0} \left(\min_{i \in N^*_{kl}} T_i^{op} + \tau^b \right) > T_0 - \tau^w \tag{3}$$

or if the second model is used and for workstation k:

$$\sum_{l=1}^{n_0} \left(\min_{i,j \in N^*_{kl}} \frac{\lambda_i}{f_j} + \tau^b \right) > T_0 - \tau^w, \tag{4}$$

then the constraint on the number of blocks for workstation k is impertinent.

Proof It is easy to see that the left-hand side of Eqs. 3 or 4 is a lower bound of the execution time on the k-th workstation. This means that real workstation time $t^w(N_k)$ cannot be less than this value and, as a consequence, if n_0 blocks are opened at this workstation, the cycle time constraint will not be respected. Therefore, it can be concluded that in any feasible solution less than n_0 blocks will be opened at workstation k. Thus, a new maximum authorized number of blocks must be set for workstation k such that Eqs. 3 or 4 will be no more satisfied. □

Using the suggested procedures, some problem instances can be easily simplified. Another interesting approach for reducing the problem size is to extract some operations. The extraction of operations is considered in the next subsection.

3.4 Operations Extraction

Taking into account two possible techniques for the block time calculation presented earlier, the following notation is used to compare the times of two operations: $T_i^{op} \leq T_j^{op}$. For the first model of the block time calculation, notation $T_i^{op} \leq T_j^{op}$ means that either $t_i \leq t_j$ or $\frac{\lambda_i}{f_i} \leq \frac{\lambda_j}{f_j}$. For the second model, it means that $\lambda_i \leq \lambda_j$ and $f_i \geq f_j$.

Proposition 8 *If $T_i^{op} \leq T_j^{op}$ and N_{kl} is the block where operation j is assigned, then the assignment of operation i to N_{kl} does not change block time $t^b(N_{kl})$.*

Proof If the first model of the block time calculation is used, then we have

$$t^b(N_{kl} \cup \{i\}) = \max\left\{\max_{h \in N_{kl}\setminus\{j\}} \frac{\lambda_h}{f_h}, \frac{\lambda_j}{f_j}, \frac{\lambda_i}{f_i}\right\} + \tau^b$$
$$\underbrace{=}_{T_i^{op} \leq T_j^{op}} \max\left\{\max_{h \in N_{kl}\setminus\{j\}} \frac{\lambda_h}{f_h}, \frac{\lambda_j}{f_j}\right\} + \tau^b = t^b(N_{kl}).$$

Similarly, for the second model we obtain

$$t^b(N_{kl} \cup \{i\}) = \frac{\max\left\{\max\limits_{h \in N_{kl}\setminus\{j\}} \lambda_h, \lambda_j, \lambda_i\right\}}{\min\left\{\min\limits_{h \in N_{kl}\setminus\{j\}} f_h, f_j, f_i\right\}} + \tau^b$$
$$\underbrace{=}_{T_i^{op} \leq T_j^{op}} \frac{\max\left\{\max\limits_{h \in N_{kl}\setminus\{j\}} \lambda_h, \lambda_j\right\}}{\min\left\{\min\limits_{h \in N_{kl}\setminus\{j\}} f_h, f_j\right\}} + \tau^b = t^b(N_{kl}).$$

Therefore, the assignment of operation i to N_{kl} does not change block time $t^b(N_{kl})$. □

Proposition 9 *If there exists $j \in \mathbf{N}$ such that $AP(j) = AS(j) = \emptyset$ and for any $e \in I^w$ we have $j \notin e$, then operation j can be removed from $\mathbf{N}$ if there exists $i \in \mathbf{N}$ such that $T_i^{op} \geq T_j^{op}$ and*

$$\left\{k : \bigcup_{l=1}^{n_0} N_{kl}^* \bigcap E(j) \neq \emptyset\right\} \bigcap \left\{k : i \in \bigcup_{l=1}^{n_0} N_{kl}^*\right\} = \emptyset, \tag{5}$$

where

$$E(j) = \bigcup_{\{e \in E^w \cup E^b : j \in e\}} e \setminus \{j\}$$

is the set of operations linked by exclusion constraints with operation j.

Proof Let N_{kl} be the block in a feasible solution where operation i is assigned. If operation j is assigned to the same block, the cycle time constraint remains respected according to Proposition 8. The precedence and inclusion constraints cannot be violated, since they do not concern operation j. The exclusion constraints hold, since no operation from $E(j)$ is assigned to workstation k because of condition 5. As a result, no problem constraint is violated by the assignment of operation j to the same block as operation i. As a consequence, operation j can be assigned to this block a posteriori and thus removed from set **N**. □

Note It is sufficient to find at least one operation i that respects the conditions of Proposition 9 to remove operation j from **N**. If there are several operations that respect these conditions, then operation j can be assigned in a final solution to the same block with any of such operations.

An operation j can be included at most in one constraint $e^* \in I^w$. Let $O_{in}(j)$ be the set of all operations with which operation j must be assigned to the same workstation:

$$O_{in}(j) = \bigcup_{i,h \in e^*} (AP(i) \cap AS(h)) \bigcup e^* \setminus \{j\}.$$

Obviously, if operation j cannot be assigned to the same workstation with an operation i, then each $h \in O_{in}(j)$ cannot be also assigned to the same workstation as operation i.

Proposition 10 *If there exists* $j \in \mathbf{N}$ *such that*

- $AP(j) = AS(j) = \emptyset$,
- $j \in e^*$, $e^* \in I^w$,

and there exists $i \in O_{in}(j)$ *such that*

1. $T_i^{op} \geq T_j^{op}$,
2. *for each* $e \in E^b$ *with* $j \in e$ *there exists* $e' = \{i, h\} \in E^b$, *where* $h \in e \setminus \{j\}$,

then j *can be deleted from* **N** *with*

- *Replacing* e^* *by* $e^* \setminus \{j\}$,
- *Replacing* j *by* i *for each* $e \in E^w$ *such that* $j \in e$,
- *Deleting all* $e \in E^b$ *such that* $j \in e$.

Proof Let N_{kl} be the block in a feasible solution where operation i is assigned. According to Proposition 8, the assignment of operation j to block N_{kl} leaves the cycle time constraint respected. The precedence constraints cannot be violated, since they do not concern operation j. The initial inclusion constraint e^* will be respected, since all $h \in O_{in}(j)$ are assigned to workstation k and operation j will be assigned to the same workstation. The workstation exclusion constraints will hold because of modified constraints E^w. The block exclusion constraints will hold in accordance with condition 2 of Proposition 10. As a result, no problem constraint is violated by the assignment of operation j to the same block as operation i. As a consequence, operation j can be assigned to this block a posteriori and thus removed from set **N**.

The aggregation of operations having $AP(j) \neq \emptyset$ or $AS(j) \neq \emptyset$ was considered in [2]. □

Table 1 Operations parameters

j	1	2	3	4	5	6	7	8	9	10
λ_j	45	45	18	28	32	45	28	13	18	54
f_j	8.33	8.33	7.5	5.83	5.33	9	9.33	8.67	9	6
T_j^{op}	5.4	5.4	2.4	4.8	6	5	3	1.5	2	9

j	11	12	13	14	15	16	17	18	19	20
λ_j	20	20	14	16	16	18	23	24	15	24
f_j	10	10	9.33	8	8	9	7.67	15	15	16
T_j^{op}	2	2	1.5	2	2	2	3	1.6	1	1.5

4 Numerical Example

To illustrate the suggested algorithms, an illustrative example is given. Set **N** consists of 20 operations. Their machining parameters are given in Table 1. The precedence graph is shown in Fig. 1.

The inclusion and exclusion constraints are as follows: $I^w = \{\{16, 17, 20\}\}$, $E^b = \{\{1, 2\}, \{2, 4, 7\}, \{3, 4\}, \{4, 6\}, \{4, 8\}, \{8, 15\}, \{10, 11\}, \{16, 17\}\}$, $E^w = \{\{5, 11\}, \{5, 12\}, \{13, 17\}, \{14, 17\}, \{18, 19\}\}$. Other problem parameters are as follows: $\tau^w = 7$, $\tau^b = 3$, $T_0 = 25$, $m_0 = 3$, $n_0 = 3$, $C_1 = 5{,}000$, $C_2 = 3{,}000$.

The application of procedures presented in Section 3.1 does not find any data incompatibility. The application of the pre-processing methods suggested in Section 3.2 lets undertake two modifications. According to Proposition 4, operation 4 can be deleted from constraint $\{2, 4, 7\} \in E^b$. And according to Proposition 5, constraint $\{13, 17\}$ can be deleted from E^w.

To apply the algorithms proposed in Section 3.3, the values of function $d(i, j)$ are calculated. The result is shown in Table 2, where the empty cells contain 0. Algorithm 1 is then run to compute q_j^-. It gives the results presented in Table 3.

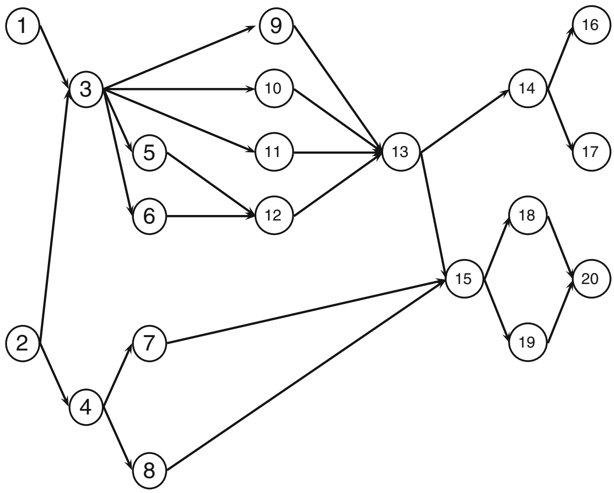

Fig. 1 Precedence graph

Table 2 Values of $d(i, j)$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	–	1																		
2	1	–					1													
3			–	1																
4			1	–		1		1												
5					–						2	2								
6				1		–							2							
7		1					–													
8				1				–							1					
9									–											
10										–	1									
11					2					1	–									
12					2							–								
13						2							–							
14														–			2			
15								1							–					
16																–	1			
17														2		1	–			
18																		–	2	
19																		2	–	
20																				–

Table 3 $[q_j^-, q_j^+]$, $j \in \mathbf{N}$ obtained with Algorithms 1 and 2

j	1	2	3	4	5	6	7	8	9	10
q_j^-	1	1	2	1	2	2	2	2	2	2
q_j^+	3	3	3	4	3	3	6	5	6	6

j	11	12	13	14	15	16	17	18	19	20
q_j^-	2	4	4	4	4	7	7	4	4	7
q_j^+	6	6	6	6	6	9	9	9	9	9

Treating the workstation inclusion constraints, Algorithm 1 has changed only the value of q_{16}^- from 4 to 7. Then, the values of q_j^+ are calculated with Algorithm 2. The obtained results are given in Table 3 as well.

According to Table 3, sets $O(q)$ are as follows: $O(1) = \{1, 2, 4\}$, $O(2) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$, $O(3) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$, $O(4) = \{4, 7, 8, 9, 10, 11, 12, 13, 14, 15, 18, 19\}$, $O(5) = \{7, 8, 9, 10, 11, 12, 13, 14, 15, 18, 19\}$, $O(6) = \{7, 9, 10, 11, 12, 13, 14, 15, 18, 19\}$, $O(7) = \{16, 17, 18, 19, 20\}$, $O(8) = \{16, 17, 18, 19, 20\}$, $O(9) = \{16, 17, 18, 19, 20\}$.

Algorithm 3 is applied and changes the value of q_2^+ from 3 to 2, since

$$\begin{aligned}\sum_{q=W(q_2^+)}^{q_2^+-1} \min_{i\in O(q)} \left\{T_i^{op} + \tau^b\right\} &= \min_{i\in O(1)} \left\{T_i^{op} + \tau^b\right\} + \min_{i\in O(2)} \left\{T_i^{op} + \tau^b\right\} \\ &= 4.8 + 3 + 1.5 + 3 = 12.3 > T_0 - \tau^w - \tau^b - T_2^{op} \\ &= 25 - 7 - 3 - 5.4 = 9.6,\end{aligned}$$

the value of q_5^+ from 3 to 2, since

$$\begin{aligned}\sum_{q=W(q_5^+)}^{q_5^+-1} \min_{i\in O(q)} \left\{T_i^{op} + \tau^b\right\} &= \min_{i\in O(1)} \left\{T_i^{op} + \tau^b\right\} + \min_{i\in O(2)} \left\{T_i^{op} + \tau^b\right\} \\ &= 4.8 + 3 + 1.5 + 3 = 12.3 > T_0 - \tau^w - \tau^b - T_5^{op} \\ &= 25 - 7 - 3 - 6 = 9,\end{aligned}$$

the value of q_6^+ from 3 to 2, since

$$\begin{aligned}\sum_{q=W(q_6^+)}^{q_6^+-1} \min_{i\in O(q)} \left\{T_i^{op} + \tau^b\right\} &= \min_{i\in O(1)} \left\{T_i^{op} + \tau^b\right\} + \min_{i\in O(2)} \left\{T_i^{op} + \tau^b\right\} \\ &= 4.8 + 3 + 1.5 + 3 = 12.3 > T_0 - \tau^w - \tau^b - T_6^{op} \\ &= 25 - 7 - 3 - 5 = 10,\end{aligned}$$

Table 4 $[q_j^-, q_j^+]$, $j \in \mathbf{N}$ modified by Algorithm 3

j	1	**2**	3	4	**5**	**6**	7	8	9	**10**
q_j^-	1	1	2	1	2	2	2	2	2	**4**
q_j^+	3	**2**	3	4	**2**	**2**	6	5	6	**5**

j	11	12	13	14	15	16	17	18	19	20
q_j^-	2	4	4	4	4	7	7	4	4	7
q_j^+	6	6	6	6	6	9	9	9	9	9

the value of q_{10}^- from 2 to 4, since

$$\sum_{q=W(q_{10}^-)}^{q_{10}^- -1} \min_{i \in O(q)} \left\{T_i^{op} + \tau^b\right\} = \min_{i \in O(1)} \left\{T_i^{op} + \tau^b\right\}$$

$$= 4.8 + 3 = 7.8 > T_0 - \tau^w - \tau^b - T_{10}^{op} = 25 - 7 - 3 - 9 = 6,$$

and the value of q_{10}^+ from 6 to 5, since

$$\sum_{q=W(q_{10}^+)}^{q_{10}^+ -1} \min_{i \in O(q)} \left\{T_i^{op} + \tau^b\right\} = \min_{i \in O(4)} \left\{T_i^{op} + \tau^b\right\} + \min_{i \in O(5)} \left\{T_i^{op} + \tau^b\right\}$$

$$= 1 + 3 + 1 + 3 = 8 > T_0 - \tau^w - \tau^b - T_{10}^{op}$$

$$= 25 - 7 - 3 - 9 = 6.$$

All these modifications are shown in Table 4. According to these modifications, sets $O(q)$ are as follows: $O(1) = \{1, 2, 4\}$, $O(2) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 11\}$, $O(3) = \{1, 3, 4, 7, 8, 9, 11\}$, $O(4) = \{4, 7, 8, 9, 11, 12\}$, $O(5) = \{7, 8, 9, 10, 11, 12, 13, 14, 15, 18, 19\}$, $O(6) = \{7, 9, 11, 12, 13, 14, 15, 18, 19\}$, $O(7) = \{16, 17, 18, 19, 20\}$, $O(8) = \{16, 17, 18, 19, 20\}$, $O(9) = \{16, 17, 18, 19, 20\}$. Due to Algorithm 3, the position of operations 5 and 6 can be defined: they must be necessary assigned to the second block of the first workstation.

5 Conclusions

The problem of equipment location in machining serial lines has been studied. This problem is known to be $\mathcal{NP}$-hard and, as a consequence, the required solution time exponentially increases with the size of the problem. A number of pre-processing procedures have been proposed in order to decrease the problem size and simplify its solution. These procedures are based on the problem constraints analyzing. Such an analysis allows deleting some constraints, transforming other ones and even extracting certain operations. The future works will concern extending this approach for other optimization problems appearing while designing machining lines.

References

1. Adam, F., Humphreys, P., eds.: Encyclopedia of Decision Making and Decision Support Technologies. Information Science Reference (2008)
2. Battaïa, O., Dolgui, A.: Reduction approaches for a generalized line balancing problem. Comput. Oper. Res. **39**(10), 2337–2345 (2012)
3. Capacho, L., Pastor, R., Dolgui, A., Guschinskaya, O.: An evaluation of constructive heuristic methods for solving the alternative subgraphs assembly line balancing problem. Journal of Heuristics **15**(2), 109–132 (2009)
4. Dolgui, A., Finel, B., Guschinsky, N., Levin, G., Vernadat, F.: MIP approach to balancing transfer lines with blocks of parallel operations. IIE Trans. **38**(10), 869–882 (2006)
5. Dou, J., Dai, X., Meng, Z.: Precedence graph-oriented approach to optimise single-product flow-line configurations of reconfigurable manufacturing system. Int. J. Comput. Integr. Manuf. **22**(10), 923–940 (2009)
6. Finel, B.: Structuration de lignes d'usinage: méthodes exactes et heuristiques. PhD thesis, Université de Metz (2004)
7. Guschinskaya, O.: Vers une optimisation de la configuration de systèmes d'usinage à boîtiers multibroches. Journal Européen des Systèmes Automatisés **44**(7), 771–790 (2010)
8. Guschinskaya, O., Dolgui, A.: Comparison of exact and heuristic methods for a transfer line balancing problem. Int. J. Prod. Econ. **120**(2), 276–286 (2009)
9. Guschinskaya, O., Dolgui, A., Guschinsky, N., Levin, G.: Minimizing makespan for multi-spindle head machines with a mobile table. Comput. Oper. Res. **36**(2), 344–357 (2007)
10. Guschinskaya, O., Dolgui, A., Guschinsky, N., Levin, G.: A heuristic multi-start decomposition approach for optimal design of serial machining lines. Eur. J. Oper. Res. **189**(3), 902–913 (2008)
11. Guschinskaya, O., Gurevsky, E., Dolgui, A., Eremeev, A.: Metaheuristic approaches for the design of machining lines. Int. J. Adv. Manuf. Technol. **55**(1–4), 11–22 (2011)
12. Ko, J., Hu, J.: Balancing of manufacturing systems with complex configurations for delayed product differentiation. Int. J. Prod. Res. **46**(15), 4285–4308 (2008)
13. Li, S., Wang, H., Hu, S., Lin, Y., Abell, J.: Automatic generation of assembly system configuration with equipment selection for automotive battery manufacturing. J. Manuf. Syst. **30**(4), 188–195 (2011)
14. Masood, S.: Line balancing and simulation of an automated production transfer line. Assem. Autom. **26**(1), 69–74 (2006)
15. Moon, I., Logendran, R., Lee, J.: Integrated assembly line balancing with resource restrictions. Int. J. Prod. Res. **47**(19), 5525–5541 (2009)
16. Nazarian, E., Ko, J., Wang, H.: Design of multi-product manufacturing lines with the consideration of product change dependent inter-task times, reduced changeover and machine flexibility. J. Manuf. Syst. **29**(1), 35–46 (2010)
17. Nof, S. (ed.): Springer Handbook of Automation. Springer (2009)
18. Özdemir, R., Ayağ, Z.: An integrated approach to evaluating assembly-line design alternatives with equipment selection. Prod. Plan. Control **22**(2), 194–206 (2011)
19. Pekin, N., Azizoglu, M.: Bi criteria flexible assembly line design problem with equipment decisions. Int. J. Prod. Res. **46**(22), 6323–6343 (2008)
20. Rose, S.: Multi-spindle machining—a fast way to make a part. Tool. Prod. (7) (2000)