Machining lines with multi-spindle workstations: a new optimization problem

Alexandre Dolgui, Ivan Ihnatsenka

MACHINING LINES WITH MULTI-SPINDLE WORKSTATIONS: A NEW OPTIMIZATION PROBLEM

A.Dolgui
Ecole des Mines de Saint-Etienne
158 cours Fauriel, 42023 Cedex 2, France
dolgui@emse.fr

I.Ihnatsenka
Y.Kupala Grodno State University
22 Ozshechko St., 230023, Belarus
ignatenko@grsu.by

Abstract

This paper deals with a transfer line optimal design. In transfer lines, operations of the same block are executed simultaneously. Blocks are assigned to machines and they can be activated in mixed order. The set of all available blocks is given beforehand. The line investment cost is defined by the sum of blocks costs and stations costs. In addition to the standard line balancing problem, precedence and cycle time constraints, blocks compatibility and parallelism constraints must be taken into account. The problem is to assign all operations grouped into blocks that all constraints are respected and line investment cost is minimum. This paper is focused on solving the problem by a branch-and-bound algorithm. A new approach for obtaining a lower bound is offered. It is based on a reduction of the transfer line balancing problem to a set partitioning problem. Computational experiments provides that the proposed approach is efficient to solve practical transfer line design problems.

1. Introduction

Production systems are often organized as an automated flow line. It allows to increase the production rate and minimize the production cost. In these lines, a product sequentially passes throughout all stations with a constant cadence. The maximal available work time per station (maximal time which product can spend at each station) is limited by a given *cycle time*. The *line cycle time* is defined by the slowest station of a line.

An important problem of a flow line design is line balancing. Historically, line balancing problem has been considered in the assembly environment. In a standard assembly line balancing problem, there is a type of product and all the operations are known. Operations must be assigned in stations such that the cycle time and the *precedence constraints* are respected and the idle time is minimum. Such single product problem is referred as the Simple Assembly Line Balancing Problem Type I (SALBP-I). For SALBP-I, the idle time is minimum iff (if and only if) the stations number is minimum as well [20].

For first time, as a combinatorial problem, SALBP has been studied in [19]. Recently, many heuristics, metaheuristics and optimal approaches have been suggested. The comprehensive surveys on SALBP are presented in [2], [14], [18], [12], [20].

There are generalizations of SALBP to multi-product systems [15], [18], [20]. An other generalization of SALBP is a cost-oriented SALBP [1] and line balancing with equipment selection [4], [3], [15]. In cost-oriented SALBP, the objective function is to minimize the cost per product unit. The goal of line balancing with equipment selection is defined as minimizing of a total cost composed of equipment, tools, equipment usage and gripper exchange costs.

This paper deals with a line balancing problem in machining/process environment which is called Transfer Line Balancing Problem (TLBP). Its main feature is that the operations to be executed are grouped into blocks. The operations of each block are executed simultaneously by one spindle head. Transfer lines have a common transfer system (a conveyor belt). The movement of product items is synchronized. There are no buffers in between stations. When item is loaded on a station, it is positioned and station spindle heads are activated in a fixed order. A typical scheme of a transfer line is presented at Figure 1. The advantage of transfer lines is they allow to essentially decrease the number of equipment pieces and line cycle time [16], [17].

In Figure 1 a station is defined by the part position and all the subsequent spindle heads. Several blocks can be installed at each station. Each spindle head is equipped by several tools. Each tool executes one operation or several operation (a combined tool).

Transfer lines are designed for mass production of single and comparatively simple product for a long exploitation time and huge production volume. Transfer lines represent "high automation" and they have high investment cost. However, their productivity allows to decrease their exploitation cost.

The set of all available blocks is often given beforehand. In this case, a transfer line is modular and is composed of "standard" spindle heads grouped in so-called

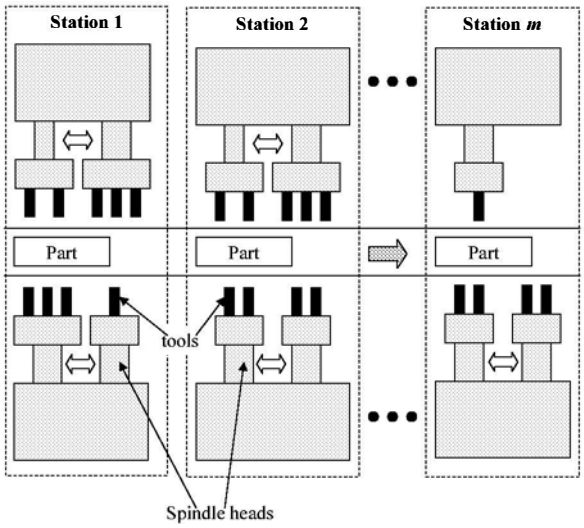

Figure 1. A scheme of a transfer line

multi-heads machines. This approach provides some flexibility when a line is redesigned (as far as possible for transfer lines) and has a great importance in many industries. The line investment cost can be estimated by a sum of stations cost and blocks costs. The goal at the line design stage is to minimize the investment cost. When the set of all available spindle heads is not given beforehand (the structure of blocks is a decision variable), the objective function is typically formulated as to minimize weighted sum of stations and blocks numbers. In [7], [8], [6], [5] it is assumed that the set of all available spindle heads is not known beforehand. The objective function is to minimize a weighted sum of stations and blocks numbers. Papers [9], [11] deal with TLBP where the set of all available spindle heads is known. In [9] blocks at each station are executed simultaneously, in [11] blocks of each station are executed sequentially. Several procedures for TLBP have been suggested: a constrained shortest path in a special digraph, mixed integer programming (MIP), branch-and-bound procedure, constraints programming and heuristics.

This paper deals with an unexplored TLBP where the set of all available blocks is given and blocks at each stations can be activated in a mixed order. The objective function is to minimize the sum of stations and blocks costs (line investment cost) while precedence, operations zoning, blocks exclusion and blocks parallelism constraints are respected.

This problem cannot be directly solved with known approaches suggested for SALBP, cost-oriented SALB and line balancing with equipment selection for the following reasons:

- The operations are partitioned into blocks.
- Operations of the same blocks are executed simultaneously by one spindle heads.
- Several available blocks can contain the same operation and it is not known which block is best to be chosen.
- The given set of available blocks commonly contains mutually incompatible blocks.
- Blocks assigned to the same station can be activated in the mixed order.
- The line cost is estimated as the sum of stations cost and blocks costs and many additional constraints are taken into account.

In this paper, we present an adaptation of the suggested in [11] approach for the problem with mixed order of blocks activation. Several procedures for improving the accuracy of a lower bound is also proposed.

2 Notations and Problem Statement

The preliminary design stage for the machining transfer lines is considered. It is supposed that the set of all operations which must to be executed is known. The goal is to define an optimal structure of the transfer line, i.e. the set of spindle heads at each stations and their activation order.

The set of all available blocks is given beforehand. In practice, the set of blocks is obtained by experience. It must contain all operations to be executed. Usually, each operation can belong to several blocks. At each station, it's blocks can be activated in a mixed order (sequentially or simultaneously). It is assumed that block execution times and costs are given. In general, for the same operation it's processing time can differ for different blocks. The detailed explanation of block time calculation is given in [8].

The transfer line balancing problem at hand is to chose and assign blocks of parallel operations in such a way that:

- Each operation is executed once.
- The given line cycle time is not exceeded.
- All operations execution satisfies to the precedence constraints.
- The machining process can cause that some groups of operations must be carried out on the same station. It is implied that the operations of each group must belong to the same block or to different blocks assigned to the same station. This type of constraint is called *operations zoning constraint.* It is assumed all these groups are defined at the product design stage and they are given.
- Similarly, technological constraints usually define sets of blocks which cannot be assigned to the same station. This type of constraint is called *blocks exclusion constraint.* In practice, these sets of blocks are

known before the optimization of a line investment cost.

- Possible sets of blocks which can be executed simultaneously are given. This type of constraint is referred as *blocks parallelism constraints.*
- For each station, the number of assigned blocks does not exceed a given value.
- The line investment cost estimated as the sum of blocks cost and stations cost is as small as possible (cost minimization).

Notations:

$\mathbf{N} = \{1, 2, \ldots, n\}$ is the given set of all the operations.

T_0 is the given transfer line cycle time.

n_0 is the maximum number of blocks for a station.

q_0 is the number of all available blocks.

$\mathbf{B} = \{b_1, b_2, \ldots, b_{q_0}\}$ is the given set of all available blocks.

$\mathcal{N}(b) \subseteq \mathbf{N}$ is the set of operations contained in a block b.

$\mathcal{N}(S) = \bigcup_{b \in S} \mathcal{N}(b)$ is a set of operations covered by set $S \subseteq \mathbf{B}$.

$t(b)$ is the processing time of block $b \in \mathbf{B}$.

$c(b)$ is the cost of block $b \in \mathbf{B}$.

C_0 is the cost of one station.

$\mathcal{P}(b)$ is the set of operations which *directly* precede (in the ordinary sense) to each operation from $\mathcal{N}(b)$. In other words, $\mathcal{P}(b)$ is a set of operations which directly precede to a block b.

$\mathcal{P}(S) = \bigcup_{b \in S} \mathcal{P}(b)$ is the set of operations which *directly* precede (in the ordinary sense) to $S \subseteq \mathbf{B}$.

$\mathcal{P}^+(b)$ is the set of operations which precede to $\mathcal{N}(b)$ directly or not. Clear, $\mathcal{P}(b) \subseteq \mathcal{P}^+(b)$.

$\mathcal{P}^+(S) = \bigcup_{b \in S} \mathcal{P}^+(b)$ is the set of operations which precede to $S \subseteq \mathbf{B}$ directly or not. Evidently, $\mathcal{P}(S) \subseteq \mathcal{P}^+(S)$.

$\mathcal{S}^+(b)$ is the set of operations which succeed to $b \in \mathbf{B}$.

$\mathcal{S}^+(S)$ is the set of operations which succeed to $S \subseteq \mathbf{B}$.

$G^{\mathcal{P}} = (\mathbf{N}, D)$ is an acyclic digraph representing the precedence constraints for operations, $(i, j) \in D$ iff operation i directly precedes (in the ordinary sense) operation j.

$G^{BE} = (\mathbf{B}, E^{BE})$ is a graph representing the blocks exclusion constraints. $(b', b'') \in E^{BE}$, $b', b'' \in \mathbf{B}$ iff b' and b'' cannot be assigned to the same station.

$G^o = (\mathbf{N}, E^o)$ is a graph representing the operations inclusion constraints. $(i, j) \in E^o$, $i, j \in \mathbf{N}$ iff i and j must be assigned to the same station. Also, the operations inclusion constraints can be defined by sets $I^o(i) = \{j \mid j \in \mathbf{N}, \text{ iff } (i, j) \in E^o\}$, $i \in \mathbf{N}$.

$G^{BP} = (\mathbf{B}, E^{BP})$ is a graph representing the blocks parallelism constraints. $(b', b'') \in E^{BP}$, $b', b'' \in \mathbf{B}$ iff b' and b'' can be simultaneously executed at the same station.

$\sqsubseteq$ is a binary relation. Let $G = (V, E)$ is a graph and $B \subseteq V$. Notation $B \sqsubseteq G$ means that all elements from B belong to the same clique in the graph G.

den(G) is a ration between the number of edges (arcs) in the graph (digraph) G and number of edges (arcs) in the complete graph (digraph) with the same number of vertices.

m is the stations number in a solution.

m^- is the lower bound of the minimum stations number.

q_k, $k = 1, 2, \ldots, m$ is the number of the sets of simultaneously activated blocks assigned to stations k in a solution.

An assignment of blocks at the station with index k is represented by an ordered family $F_k = (S_1^k, S_2^k, \ldots, S_{q_k}^k)$, where: $S_u^k \subseteq \mathbf{B}$, $k = 1, 2, \ldots, m$, $u = 1, 2, \ldots, q_k$ is a set of simultaneously activated blocks, q_k is a number of such sets assigned to the station k. Index u indicates the order of the sequential execution for sets S_u^k.

The considered transfer line design problem is stated as follows: to find a family $\mathcal{L} = (F_1, F_2, \ldots, F_m)$ respected the following constraints.

All operations are executed:

$$\bigcup_{k=1}^{m} \Big(\bigcup_{S \in F_k} \mathcal{N}(S) \Big) = \mathbf{N}. \tag{1}$$

Each operation is executed once:

$$\begin{gathered} \mathcal{N}(b') \bigcap \mathcal{N}(b'') = \varnothing,\ b' \in S_u^k,\ b'' \in S_v^r, \\ b' \neq b'',\ k, r = 1, 2, \ldots, m, \\ u = 1, 2, \ldots, q_k,\ v = 1, 2, \ldots, q_r. \end{gathered} \tag{2}$$

The operations precedence constraints are respected:

$$\begin{gathered} \text{for all } k = 1, 2, \ldots m \text{ and for all } S_u^k \in F_k \\ \mathcal{P}(S_u^k) \subseteq \Big(\bigcup_{r=1}^{k-1} \bigcup_{v=1}^{q_r} \mathcal{N}(S_v^r) \Big) \bigcup \Big(\bigcup_{v=1}^{u} \mathcal{N}(S_v^k) \Big). \end{gathered} \tag{3}$$

All blocks from set S_u^k are executed simultaneously $k = 1, 2, \ldots, m$, $u = 1, 2, \ldots, q_k$. Then, the set E^{BP}

must contain all edges (b', b''), $b' \neq b''$, $b', b'' \in S_u^k$. In other words, S_u^k is a clique in the G^{BP}. So, blocks parallelism conditions can be formulated as:

$$S_u^k \sqsubseteq G^{BP},\ k = 1, 2, \ldots, m,\ u = 1, 2, \ldots, q_k. \quad (4)$$

Blocks exclusion constraints:

$$\begin{gathered}(b', b'') \notin E^{BE},\ \text{for all } b', b'' \in \bigcup_{u=1}^{q_k} S_u^k, \\ k = 1, 2, \ldots, m,\ u = 1, 2, \ldots, q_k.\end{gathered} \quad (5)$$

Each station time does not exceed the given objective line cycle time:

$$\sum_{S \in F_k} \max_{b \in S} t(b) \leq T_0, \quad k = 1, 2, \ldots, m. \quad (6)$$

For each station, the number of assigned blocks does not exceed the given value:

$$\sum_{S \in F_k} \left|S\right| \leq n_0, \quad k = 1, 2, \ldots, m. \quad (7)$$

Operations zoning constraints:

$$\begin{gathered}\bigcup_{S \in F_k} \bigcup_{b \in S} \bigcup_{i \in \mathcal{N}(b)} I^o(i) \subseteq \bigcup_{S \in F_k} \mathcal{N}(S), \\ k = 1, 2, \ldots, m.\end{gathered} \quad (8)$$

The line investment cost is as small as possible:

$$\min C(\mathcal{L}) = \sum_{k=1}^{m} \sum_{S \in F_k} \sum_{b \in S} c(b) + C_0 m. \quad (9)$$

The decision variables are sets S_u^k grouped into families F_k, $k = 1, 2, \ldots, m$, $u = 1, 2, \ldots, q_k$.

3 Lower bound

In [10], a TLBP with simultaneously activated blocks has been investigated. Because that at each station assigned blocks are executed simultaneously the line cycle time constraints are neglected. It was suggested a MIP approach to solve this problem. Computational experiments have been done by CPLEX. They showed that CPLEX can solve in appropriate time only problems where number of operations does not exceed 40. It is generally known that CPLEX solves MIP problem by a branch-and-bound algorithm. It obtains a lower bound by the linear-programming relaxing of the initial MIP problem. In fact, a linear-programming lower bound is insufficient to solve complex transfer line balancing problem.

In this paper, it is suggested to obtain a lower bound by relaxing of the initial TLBP to a special set partitioning problem. This approach allows to calculate lower bound more precisely although it is more time-consuming.

Let $\mathcal{L} = (F_1, F_2, \ldots, F_m)$ is family which satisfies constraints (2)–(7) and constraint (8) for $k = 1, 2, \ldots, m-1$. Evidently, $\mathcal{L}$ represents itself a *partial solution* of the problem (1)–(9). The constraint (1) is not respected for $\mathcal{L}$. Otherwise, it is not necessary to find a lower bound. Denote

$$\Psi_{\mathcal{L}} = \bigcup_{F_k \in \mathcal{L}} \bigcup_{S \in F_k} \mathcal{N}(S). \quad (10)$$

The set $\Psi_{\mathcal{L}}$ can be considered as a *state* of design. Let

$$\Xi = \bigcup_{u=1}^{q_m} S_u^m,\ \overline{\Psi}_{\mathcal{L}} = \left(\mathbf{N} \setminus \Psi_{\mathcal{L}}\right) \bigcup \mathcal{N}(\Xi), p' = \sum_{u=1}^{q_m} \left|S_u^m\right|.$$

Assume, $\mathcal{W} = (w_1, w_2, \ldots, w_p)$ is a set of blocks satisfying the constraints:

$$\begin{gathered}w_k \in \Xi,\ k = 1, 2, \ldots, p', \\ w_k \in \mathbf{B},\ k = p' + 1, p' + 2, \ldots, p,\end{gathered} \quad (11)$$

$$\mathcal{N}(w_k) \subseteq \overline{\Psi}_{\mathcal{L}}, \quad k = 1, 2, \ldots, p, \quad (12)$$

$$\begin{gathered}\mathcal{N}(w_k) \bigcap \mathcal{N}(w_u) = \varnothing, \\ k \neq u,\ k, u = 1, 2, \ldots, p,\end{gathered} \quad (13)$$

$$\mathcal{N}(\mathcal{W}) = \overline{\Psi}_{\mathcal{L}}, \quad (14)$$

If the first $m-1$ stations of $\mathcal{L}$ are completely determined, then the family $\mathcal{W}$ can be considered as a "complement" to the partial solution $(F_1, F_2, \ldots, F_{m-1})$. As follows from the definition, elements w_k, $k = 1, 2, \ldots, p'$ are defined accordingly to the last station in the partial solution $\mathcal{L}$.

The investment costs of a subfamily $(F_1, F_2, \ldots, F_{m-1}) \subseteq \mathcal{L}$ is equal to:

$$\alpha = \sum_{k=1}^{m-1} \sum_{S \in F_k} \sum_{b \in S} c(b) + C_0(m-1). \quad (15)$$

Evidently, the value α is a constant for the given $\mathcal{L}$.

A lower bound of the investment costs of assigning the set $\mathcal{W}$ can be estimated as:

$$\beta(\mathcal{W}) = \sum_{k=1}^{p} c(w_k) + C_0 \widetilde{m}, \quad (16)$$

where $\widetilde{m} = \widetilde{m}(\mathcal{W})$ is a lower bound of the stations number which are needed to assign the set of blocks $\mathcal{W}$.

Thus, a lower bound of the total investment cost resulted from assigning the subfamily $(F_1, F_2, \ldots, F_{m-1})$ and the set $\mathcal{W}$ is defined as:

$$LB(\mathcal{W}) = \alpha + \beta(\mathcal{W}). \quad (17)$$

Let there exists a set $\mathcal{W}^* = (w_1, w_2, \ldots, w_{p'}, w_{p'+1}^*, w_{p'+2}^*, \ldots, w_{p^*}^*)$ satisfying constraints (11)–(14) and minimizing function (16). Then, the value:

$$LB_{\mathcal{L}} = \alpha + \beta(\mathcal{W}^*) = \alpha + \min_{\mathcal{W}} \beta(\mathcal{W}) \quad (18)$$

can be considered as a lower bound of the investment cost for the partial solution $\mathcal{L}$.

The constraints (13), (14) appear in the so-called set partitioning problem. It was studied in [13]. Several approaches based on back-tracking, integer linear and dynamical programming were proposed to solve the set partitioning problem.

The problem (11)–(14), (18) differs from the standard set partitioning problem by the element $C_0\widetilde{m}$ (see formula (16)) In general, function $\widetilde{m}(\mathcal{W})$ depends on the graphs G^{BE}, G^{BP}. Hence, it is nonlinear and the problem (11)–(14), (18) is essentially complex than the standard set partitioning problem.

3.1 Lower bound of the stations number

Suppose, it is necessary to estimate the number of the station $\widetilde{m}$ for a set $\mathcal{W}$ satisfying constraints (11)–(13), and, may be, constraint (14).

Consider blocks exclusion constraint (5). Let $\overline{G}^{BE}$ be the complement of the graph G^{BE}. A set $\mathcal{B} \subseteq \mathbf{B}$ induces in the graph $\overline{G}^{BE}$ a subgraph. Let (V_z, E_z), $z = 1, 2, \ldots, z_0$ be the components of this subgraph. Evidently, if two blocks belong to two different components, then they cannot be assigned to the same station. Hence, the estimation of $\widetilde{m}$ for the set $\mathcal{W}$ can be reduced to the estimation of the stations number $\widetilde{m}_z$ for each component (V_z, E_z), $z = 1, 2, \ldots, z_0$. The numbers $\widetilde{m}_z$ can be estimated by taking into account constraints (4)–(7).

First, consider constraints (5), (7). In each component (V_z, E_z), the blocks of a set $V \subseteq V_z$ cannot be assigned to the same station iff they belong to the same independent set of vertices in the component (V_z, E_z). Hence, the size of the maximum independent set defines the lower bound of the number $\widetilde{m}_z$. A lower bound τ^- of the size of maximum independent set α_0 for the component (V_z, E_z) can be computed by the following formula:

$$\widetilde{m}_z \geq \alpha_0(V_z, E_z) \geq \tau^- = \Big\lceil \sum_{v \in V_z} (1 + deg(v))^{-1} \Big\rceil, \quad (19)$$

where $deg(v)$ is the degree of the vertex v in the component (V_z, E_z),

On the other hand, the blocks of a set $V \subseteq V_z$ can be assigned to the same station iff they belong to the same clique at the component (V_z, E_z). An upper bound τ^+ of the size of maximum clique $\omega(V_z, E_z)$ can be determined as:

$$\tau^+ \leq \omega(V_z, E_z) = \Delta(V_z, E_z) + 1, \quad (20)$$

where $\Delta(V_z, E_z)$ is the maximum degree of the vertices in the component (V_z, E_z).

Using the formulas (19), (20) and taking into account constraint (7) a lower bound of the stations number $\widetilde{m}_z$ can be obtained as:

$$\widetilde{m}_z \geq \max\Big(\tau^-, \Big\lceil \big|V_z\big| / \min(n_0, \tau^+) \Big\rceil\Big). \quad (21)$$

Now, consider the blocks parallelism (4), cycle time (6) and maximum number of blocks constraint (7). The blocks from the set V_z must be assigned to stations in the mixed order. This condition leads to the problem: to find out the partition $X = (X_1, X_2, \ldots, X_d)$ of the set V_z satisfying the following constraints.

Blocks of set X_k belongs to the same clique in the graph G^{BP}, i.e. they are executed simultaneously:

$$X_k \sqsubseteq G^{BP},\ k = 1, 2, \ldots, d. \quad (22)$$

The execution time of each set of parallel blocks does not exceed the line cycle time:

$$\max_{b \in X_k} t(b) \leq T_0,\ k = 1, 2, \ldots, d, \quad (23)$$

The number of blocks for each set does not exceed the given value:

$$\big|X_k\big| \leq n_0,\ k = 1, 2, \ldots, d. \quad (24)$$

All blocks from the set V_z are assigned:

$$\bigcup_{k=1}^{d} X_k = V_z. \quad (25)$$

The objective function can be defined as minimization of the time required to carry out the blocks of the component (V_z, E_z), i.e.

$$t^-(V_z) = \min\ T(X) = \min_X \sum_{k=1}^{d} \max_{b \in X_k} t(b). \quad (26)$$

In general, problem (22)–(26) is a very complex combinatorial problem. However, for the TLBP the density $den(G^{BP})$ is enough small. Usually, it less than three percents and problem (22)–(26) can be solved by a tree-search algorithm with the dominance rule:

Proposition 1 *If partitions X' and X'' satisfy the constraints (22)–(24), $\bigcup_{X \in X'} X \subseteq \bigcup_{X \in X''} X$ and $T(X') \geq T(X'')$, then the partition X' is dominated by X''.*

So, the lower bound $t^-(V_z)$ of blocks execution time allows to obtain a lower bound of the stations number as:

$$\widetilde{m}_z \geq \Big\lceil t^-(V_z)/T_0 \Big\rceil. \quad (27)$$

Finally, expressions (21), (27) lead that a lower bound of the stations number for the component (V_z, E_z) can be stated as follows:

$$\widetilde{m}_z = \max\Big(\tau^-, \Big\lceil \frac{\big|V_z\big|}{\min(n_0, \tau^+)} \Big\rceil, \Big\lceil \frac{t^-(V_z)}{T_0} \Big\rceil\Big). \quad (28)$$

Hence, a lower bound of the stations number for the set $\mathcal{W}$ is equal to:

$$\widetilde{m} = \sum_{z=1}^{z_0} \widetilde{m}_z. \quad (29)$$

Last expression completely determines the calculation of objective function (17) for a given set $\mathcal{W}$.

3.2 Algorithm for obtaining the lower bound

The lower bound $LB_{\mathcal{L}}$ can be obtained by solving problem (11)–(14), (18). As mentioned above, this problem is a special set partitioning problem. In advance of [13], proposed in this paper algorithm is a frontier branch-and-bound procedure. However, it uses the reductions mentioned in [13] in each node of search-tree.

Each set $\mathcal{W}$ satisfies constraints (11)–(13). A lower bound $LB^-(\mathcal{W})$ of function (17) can be defined as following:

$$LB^-(\mathcal{W}) = \alpha + \sum_{w\in\mathcal{W}} c(w) + C_0\frac{\sum_{z=1}^{z_0} t^-(V_z)}{T_0} + \hat{h}\big|\overline{\Omega}\big|,$$

where $\hat{h}$ represents the minimum average "contribution" of each non-assigned operation to function (17). For the partial solution $\mathcal{L} = \varnothing$ (it is possible only for the root lower bound) $\hat{h}$ can be defined as:

$$\hat{h} = \Big(\sum_{i\in\mathbf{N}} \min_{b\in\mathbf{B}, i\in b} \frac{c(b)}{\big|\mathcal{N}(b)\big|} + C_0 m^-\Big)/\big|\mathbf{N}\big|, \qquad (30)$$

where m^- is the minimum number of stations m^-, without taking into account the stations and blocks cost. Number m^- can be estimated in different ways. For example, it can be done before line cost optimization by a tree-search algorithm. In the cases when the partial solution $\mathcal{L} \neq \varnothing$ the value of $\hat{h}$ can be estimated more precisely:

$$\hat{h} = LB_{\mathcal{L}_p}/\big|\mathbf{N}\big|, \qquad (31)$$

where $\mathcal{L}_p$ is such a partial solutions that $\mathcal{L}$ can be obtained by adding of some blocks to $\mathcal{L}_p$. If problem (1)–(9) is solved by a branch-and-bound algorithm, then it is reasonable to suppose $LB_{\mathcal{L}_p}$ to be equal to a lower bound of the "parent" node. Computational experiments are shown the branch-and-bound procedure efficiently solves problem (11)–(14), (18) (see Tables 1).

4 Branch-and-Bound Algorithm

Suppose that the first $m-1$ stations in the current partial solution $\mathcal{L}$ are completely determined. Each partial solution (family) is associated with a node of the search tree. The node extension is a composition of a current partial solution and such a block that a new partial solution satisfies to constraints (2)–(7) and constraint (8) for $k = 1, 2, \ldots, m-1$. Denote FB is a set of blocks satisfying the following constraints:

$$FB = \{b \,|\, \mathcal{N}(b) \cap \Psi_{\mathcal{L}} = \varnothing, \mathcal{P}(b) \subseteq \Psi_{\mathcal{L}} \cup \mathcal{N}(b)\}.$$

In other words, the set FB consists of blocks such that any operation of a block has no predecessors or its predecessors are already assigned.

Generally, for a given partial solution $\mathcal{L}$, each block $b \in FB$ can generate at most three families:

1. $\mathcal{L}^c = (F_1, F_2, \ldots, F_{m-1}, (S^m_1, S^m_2, \ldots, S^m_{q_m-1}, S^m_{q_m} \cup \{b\}))$.
2. $\mathcal{L}^t = (F_1, F_2, \ldots, F_{m-1}, (S^m_1, S^m_2, \ldots, S^m_{q_m}, \{b\}))$.
3. $\mathcal{L}^n = (F_1, F_2, \ldots, F_m, (\{b\}))$.

All these families must respect constraints (2)–(7) and constraint (8) for $k = 1, 2, \ldots, m-1$. If some families do not respect these constraints, then they are infeasible and cannot be further branched.

So, the branching for the current partial solution is the generation of possible $\mathcal{L}^c$, $\mathcal{L}^t$, $\mathcal{L}^n$ families for each block $b \in FB$.

A lack of this approach is in the fact that it is possible to have many families which differs only in the order of blocks and operations. The following method based on dominance rules can be used to overcome this lack and to essentially increase the performance of the branch-and-bound algorithm.

4.1 Dominance rules

For the families $\mathcal{L}^c$, $\mathcal{L}^t$, $\mathcal{L}^n$ if there is a lower bound for one families, then there are lower bound for all three families. This conclusion is based on the procedures branch-and-bound algorithm for obtaining lower bound. Moreover, if there exist lower bounds, then families $\mathcal{L}^c$, $\mathcal{L}^t$, $\mathcal{L}^n$ are equivalent (composed of the same blocks). The definition of the families $\mathcal{L}^c$, $\mathcal{L}^t$, $\mathcal{L}^n$ and the Bellman's principle of optimality leads to:

Proposition 2 *If family $\mathcal{L}^c$ satisfies constraints (2)–(7) and constraint (8) for $k = 1, 2, \ldots, m-1$, then family $\mathcal{L}^t$ is infeasible or unpromising.*

Proposition 3 *If family $\mathcal{L}^c$ satisfies constraints (2)–(8), then families $\mathcal{L}^t$, $\mathcal{L}^n$ are infeasible or unpromising.*

Proposition 4 *If family $\mathcal{L}^t$ satisfies constraints (2)–(8), then families $\mathcal{L}^n$ is infeasible or unpromising.*

These propositions allow to eliminate some branches at the current node. Similar action must be repeated for all other blocks of the current node.

A more powerful approach for reducing the search tree is to store and use $\Psi_{\mathcal{L}}$ states throughout of the search.

If the states of two partial solutions are equal, then their $\overline{B}$ sets are equal as well.

Proposition 5 *Let a partial solution $\mathcal{L}$, satisfying constraints (2)–(8), has the state $\Psi_{\mathcal{L}}$. If there exists a family $\widehat{\mathcal{L}}$, respecting the same constraints, such that $\Psi_{\mathcal{L}} = \Psi_{\widehat{\mathcal{L}}}$, $C(\widehat{\mathcal{L}}) < C(\mathcal{L})$, then $\mathcal{L}^n$ families for any block $b \in FB$ are infeasible or unpromising.*

Last statement can be applied by storing $(\Psi_{\mathcal{L}}, C(\mathcal{L}))$ pairs (where $\mathcal{L}$ respects constraints (2)–(8) during the search). Let DS is an appropriate data structure (such as list, balanced tree, etc.) containing such $(\Psi^{DS}, C^{DS}_{\Psi})$

pairs, where Ψ^{DS} is a state and C_{Ψ}^{DS} is the best value of the objective function $C(\cdot)$. Three actions are applied to DS : *checking, adding, updating.*

The checking consists of the search in DS for the pair with the state equal to $\Psi_{\mathcal{L}}$. If DS does not contain such a pair, then the adding procedure is executed. Otherwise, the condition $C(\mathcal{L}) < C_{\Psi}^{DS}$ is tested. If it holds, then the corresponding updating must be done by the assignment $C_{\Psi}^{DS} = C(\mathcal{L})$. Otherwise, due to proposition 5 all $\mathcal{L}^n$ families for any block $b \in FB$ are eliminated from the search tree. An adding is executed when DS does not contain a pair with the state $\Psi_{\mathcal{L}}$. In this case, the pair $(\Psi_{\mathcal{L}}, C(\mathcal{L}))$ is added to the data structure DS.

The effectiveness of proposed approach depends on the number of possible states.

4.2 Design of the branch-and-bound algorithm

The proposed branch-and-bound algorithm is a standard depth first search with a multi-choice tree. The main stages of the algorithm are the following:

Stage I. Obtaining of a root lower bound: The minimum stations number m^- is estimated. Using this number, a root lower bound for $\mathcal{L} = (\varnothing)$ is computed by the branch-and-bound algorithm and formula (30). The objective function of the current best solution is supposed to be equal to ∞.

Stage II. Node extension: For the current node, a set FB is computed. Each block from FB generates new descendants of current node with partial solutions $\mathcal{L}^c$, $\mathcal{L}^t$, $\mathcal{L}^n$. These partial solutions are checked by the dominance rules (see propositions 2–5). Infeasible or unpromising nodes are pruned. For the rest of descendants, lower bounds are calculated. Descendants having the lower bound great or equal than the current best solution value are also pruned. If for a node the corresponding partial solution covers the set of all operations and its solution value is less than the current best value, then this partial solution is supposed to the best. The node having minimum lower bound is supposed to be most promising and must be branched first.

Stage III. Stop condition: If there are nodes with lower bound or solution value less than the current best value, then the search is continued from the previous stage. Otherwise, an optimal solution has been founded.

5 Numerical experiments

The aim of the experimental study is to examine the impact of various parameters on the performance of the proposed algorithm. and the average effectiveness of the lower bound. The quality of the lower bound is measured by a relative distance between the root lower bound and the optimum (gap).

Test examples were generated randomly for three values of $|\mathbf{N}|$, $|\mathbf{B}|$ and for two values of $den(G^{\mathcal{P}}) \in \{0.1, 0.15\}$ and $den(G^{BE}) \in \{0.1, 0.15\}$. Because the graphs $G^{\mathcal{P}}$, G^{BE}, G^{o} and G^{BP} densities must be coordinated, $den(G^{o})$ and $den(G^{BP})$ are fixed at the values 0.01. and 0.03, respectively.

For other parameters the following values are used: $500 \leq C_0 \leq 1000$, $n_0 = 5$, $120 \leq T_0 \leq 150$, $200 \leq c(b) \leq 280$, $20 \leq t(b) \leq 55$. All parameters are independently and uniformly distributed.

There 12 series of tests are generated, each series is composed of 25 examples with different different values of above parameters. Tests were carried out on Intel IV with 2.8Ghz and the obtained results are presented in Table 1.

The performance of the branch-and-bound algorithm is measured by total running time (preprocessing time and branching time).

The gap is computed with the formula $100\% \cdot (C^* - LB_r)/C^*$, where C^* is the optimal solution and LB_r is the corresponding root lower bound.

Table 1. Computational experiments

	$den(G^{\mathcal{P}}) = 0.1$						$den(G^{\mathcal{P}}) = 0.15$					
	$den(G^{BE}) = 0.05$			$den(G^{BE}) = 0.1$			$den(G^{BE}) = 0.05$			$den(G^{BE}) = 0.1$		
$\|\mathbf{N}\|$	50	60	70	50	60	70	50	60	70	50	60	70
Min $\|\mathbf{B}\|$	92	109	124	88	105	125	83	104	126	79	105	125
Max $\|\mathbf{B}\|$	96	117	139	96	118	135	97	118	135	96	118	134
Average $\|\mathbf{B}\|$	94	113	133	92	111	131	93	112	131	91	111	130
Min preprocessing time	15	43	64	18	57	93	21	43	93	31	56	72
Max preprocessing time	49	521	366	56	166	979	94	261	349	76	296	569
Average preprocessing time	31	195	157	37	94	247	47	119	202	50	136	238
Min total running time	35.3	74.3	220	46.4	141.6	504.6	39	65	181	49.6	145.5	213
Max total running time	785.4	8685	16722	8231	19052	25375	590	1589	2630	1224	4087	4051
Average total running time	147.1	1553	2731	680	4994	6416	119	432	1048	185	788	1432
Average lower bound time	0.005	0.019	0.021	0.008	0.018	0.048	0.011	0.028	0.056	0.01	0.037	0.06
Max gap (%)	30.8	27.0	19.5	20.3	24.8	22.6	26.3	22.6	26.8	29.6	27.9	21.0
Min gap (%)	0.0	2.5	4.2	0.0	7.2	5.4	2.2	4.6	8.0	6.8	7.9	9.8
Average gap (%)	10.8	11.2	10.8	11.2	14.6	14.7	13.1	13.8	14.2	16.8	16.1	15.4
Min improvement (%)	0.5	4.3	3.7	0.0	0.0	2.3	0.0	0.2	2.2	0.0	0.0	5.1
Max improvement (%)	17.3	17.9	15.2	27.1	18.7	20.4	21.9	16.6	23.9	20.5	25.8	14.8
Average improvement (%)	8.9	9.9	9.8	8.8	14.6	11.4	8.1	6.9	8.8	8.7	10.1	9.8

The total running time rapidly increases when the number of operations and blocks increases. The density $den(G^{\mathcal{P}})$ has a great influence as well. Namely, the total running time decreases when the density $den(G^{\mathcal{P}})$ increases. In fact, the more the density $den(G^{\mathcal{P}})$ is the less the size of search tree is.

Table 1 indicates that the average gap increases when the $den(G^{BE})$ increases as well. Because the large value of the gap rises the number of examined nodes, then the performance of the branch-and-bound algorithm is reduced down when $den(G^{BE})$ increases. The average gap for all the tests is equal to 13.6%. The moderate size of real transfer line balancing problem is 70 operations. So, the developed algorithm allows to optimally solve the most of real TLBP in three hour in average.

The relative improvement is computed with the formula $(C_1 - C^*)/C_1$, where C_1 is a first solution value obtained during the branching. Table 1 indicates the average improvement is equal 9.5%.

6 Conclusion

In this paper a new line balancing problem is considered. This problem appears during design of machining transfer lines with spindle heads activated in a mixed order. In contrast to the SALBP, considered transfer line balancing problem has many additional properties and constraints and it is more complex.

An optimization model of the problem is developed. An approach to compute a lower bound is proposed. The approach is based on the reduction of the transfer line balancing problem to a special set partitioning problem. The proposed branch-and-bound algorithm also uses dominance rules for reducing the search tree. Computational experiments indicate that the suggested approach is efficient to solve optimally problems with moderate size. Namely a problem with 70 operations can be solved in three hour. Computational experiments demonstrate the sensitivity of running time on some studied parameters.

In average, the proposed algorithm improves the line investment cost on 9.5%. This economic benefit justifies for searching an optimal solution although time consuming. But for large-scale problems the calculation time for proposed algorithm can be prohibitive. Further study can be concerned with a design of heuristic techniques based on this algorithm.

References

[1] M. Amen. Heuristic methods for cost oriented assembly line balancing: A survey. *International Journal of Production Economics*, 68:1–14, 2002.

[2] I. Baybars. A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, 32:909–932, 1986.

[3] J. Bukchin and J. Rubinovitz. A weighted approach for assembly line design with station paralleling and equipment selection. *IIE Transactions*, 35:73–85, 2002.

[4] J. Bukchin and M. Tzur. Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32:585–598, 2000.

[5] A. Dolgui, B. Finel, N. Guschinsky, G. Levin, and F. Vernadat. An heuristic approach for transfer lines balancing. *J. of Intelligent Manufacturing*, 16(2):159–171, 2005.

[6] A. Dolgui, N. Guschinsky, Y. Harrath, and G. Levin. Une approache de programmation linéaire pour la conception des lignes de transfert. *European Journal of Automated Systems (JESA)*, 36(1):11–31, 2002.

[7] A. Dolgui, N. Guschinsky, and G. Levin. On problem of optimal design of tranfer lines with parallel and sequential operations. In J. Fuertes, editor, *Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factor Automation*, volume 1, pages 329–334, Barcelona, 1999. IEEE.

[8] A. Dolgui, N. Guschinsky, and G. Levin. Appoaches to balancing of transfer line with blocks of parallel operations. Technical report, Institute of Engineering Cybernetics/ Universite of Technology Troy, Misk, Belarus, 2000.

[9] A. Dolgui, N. Guschinsky, and G. Levin. A special case of transfer lines balancing by graph approach. *European Journal of Operational Research*, page In Press, 2005.

[10] A. Dolgui, N. Guschinsky, G. Levin, M.Louly, and S. Belmokhtar. Balancing of transfer lines with simultaneously activated spindle. In G. Morel and C. Pereira, editors, *Preprints of the IFAC Symposium on Information Control Problems in Manufacturing (INCOM'04)*, pages CD–ROM, Salvador, Brazil, 2004. IFAC.

[11] A. Dolgui and I. Ihnatsenka. Branch and bound algorithm for optimal design of transfer lines with multi-spindle stations. Technical report, G2I Division, Ecole des Mines, Saint-Etienne,France, 2004.

[12] E. Erel and S. Sarin. A survey of the assembly line balancing procedures. *Production Planning and Control*, 9(5):414–434, 1998.

[13] R. Garfinkel and G. Nemhauser. The set partition problem: set covering with equality constraints. *Operations Research*, 17:848–856, 1969.

[14] S. Ghosh and R. Gagnon. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly line systems. *International Journal of Production Research*, 27:637–670, 1989.

[15] S. Graves and C. H. Redfield. Equipment selection and task assignment for multiproduct assembly system design. *The International Journal of Flexible Manufacturing Systems*, 1:31–50, 1988.

[16] M. Groover. *Automation, Production Systems and Computer Integrated Manufacturing*. Prentice Hall, Eaglewood Cliffs, New Jersey, 1963.

[17] K. Hitomi. *Manufacturing System Engineering*. Taylor&Francis, New York, 1996.

[18] B. Rekiek, A. Dolgui, A. Delchambre, and A. Bratcu. State of art of assembly lines design optimisation. *Annual Reviews in Control*, 26(2):163–174, 2002.

[19] M. Salveson. The assembly line balancing problem. *Journal of Industrial Engineering*, 6:18–25, 1955.

[20] A. Scholl. *Balancing and sequencing of assembly lines*. Physica-Varlag, Heidelberg, 1999.