Reconfigurable Transfer Lines Cost Optimization – A Linear Programming Approach

Antoneta Iuliana BRATCU[*,**]
[*]"Dunărea de Jos" University of Galați
47, Domnească
800008 – Galați, Romania
Antoneta.Bratcu@ugal.ro

Alexandre DOLGUI[**]
Sana BELMOKHTAR[**]
[**]École des Mines de Saint Étienne
158, Cours Fauriel
42023 – Saint Étienne, France
{dolgui,belmokhtar}@emse.fr

Abstract

The transfer lines dealt with in this paper are equipped with multifunctional spindle heads, capable to realize several operations in the same time (in parallel) on the same work-piece. Each such multifunctional spindle head is called a block and is fully described by its set of operations, operating time and cost. The reconfigurability concerns here the possibility of realizing a family of products with the same equipment. The blocks and the product family features are supposed known. The problem is to assign the blocks to workstations such as to minimize the total cost of the line and to meet a set of time and technological constraints for the whole product family. This problem is formulated as a linear program (LP) and solved by using the Cplex ILOG optimization software. An example is provided to illustrate the steps of the proposed solving method.

1. Introduction. Previous work

Transfer lines are automated flow-oriented production systems, typically encountered in the mechanical industry [1][2]. Their use is justified mainly for the mass production, as they increase the production rate and minimize the cost of machining parts [3]. These lines appear as a linear sequence of multi-spindle machines (workstations), without buffers in between, arranged along a conveyor belt (transfer system). Each workstation is equipped with several spindles heads, each of these latter being composed of several tools. Each tool executes one or several (combined tool) operations. The operations to be executed are grouped into the so-called *blocks*; each block contains the operations executed simultaneously by one spindle head. When all blocks of a workstation have been accomplished, the workstation cycle time is terminated. The line's production rate is the inverse of its cycle time, taken as the longest workstation cycle time.

Transfer lines are designed for a long exploitation time and need high investments. Once an optimal structure is found and implemented, it will be difficult and expensive to change it in response to product type changes. It is more likely that such a line be designed from the very beginning for all the product types intended to be produced. Research has been conducted to an integrated approach of transfer lines design in the context of *flexibility* [4]. The problem dealt with in this paper is that of designing modular *reconfigurable* transfer lines which are composed of standard spindle heads and required to produce a *family of similar products*. For simplicity, the spindle heads will be here called blocks.

The features of the product family are supposed known, meaning that each product is given by the corresponding precedence graph and the desired cycle time. The products must necessarily resemble in some admissible degree, as to some well known rules of defining a product family [5] (for example, they must have a minimum number of common operations). The transfer line configuration, minimal from the cost point of view, must be defined for an imposed output.

The following assumptions are adopted for the whole family:

– a set of all possible blocks is known and it is authorized that each block may be used only partially; the operations from a block are executed in parallel;

– each operation must be executed only once, by a single block assigned to exactly one workstation;

– blocks performing inside of a station is sequential;

– station setting cost, blocks' operating times and cost of each block are given.

The problem is to find an admissible assignment of blocks to stations in such a way that all the technological constraints – order, compatibility and desired cycle times – be satisfied and the *total cost* of all stations and blocks be the minimal possible. To find the blocks activation sequence for each product is not our goal here; instead, we must ensure that the provided solution allows that such an order always exist for each product.

This kind of problems is known in literature as *Line Balancing* problems [6]. If considering a single product type and supposing that each block is composed of only

0-7803-9402-X/05/$20.00 © 2005 IEEE

one operation, then our problem is reduced to the basic problem, *Simple Assembly Line Balancing* (SALB), whose goal is to minimize the unbalance (cost) of the line for a given line cycle time. The unbalance is minimal if and only if the number of stations is minimal.

Generally, SALB integer linear programming models are formulated and solved by exact or heuristic methods [7][8][9][10][11]. The *Mixed-model Assembly Line Balancing* (MALB) problem approaches the optimization of lines with several "versions" of a commodity in an intermixed sequence [12]. This problem takes into account the possible differences between versions – among others, different precedence relations, different task times, etc. *Generalized* ALB (GALB) problems have also been stated, that enriches the basic assumptions of SALB, in order to solve more realistic problems – a comprehensive survey may be found in [13].

The balancing problems with *equipment selection* have some common features with the studied problem. But our problem cannot be directly solved by the known methods for some essential distinctions versus the balancing and equipment selection problems. The most important is the parallelization of operations into blocks.

The problems the closest of ours are studied in [14][15][16][17][18], where all blocks of parallel operations at the same station are executed sequentially (block by block) and any alternative variants of blocks are not given beforehand (any subset of the given operation set can be considered as a block). In [19][20][21] the blocks are known and they are executed in parallel. All these papers concern the case of a single product. Three approaches have been proposed: a constrained shortest path; mixed integer programming (MIP); heuristics. This paper aims at generalizing the linear programming approach to the case of a product family.

The paper is organized as follows. Section 2 describes the problem in details, introduces notations and explains how the constraints' aggregation is made. In Section 3 the linear programming formulation is presented and some model reductions are proposed. Section 5 is dedicated to discussion over a small scale example. Finally, conclusions and future points are listed in Section 5.

2. Problem description

2.1. Input data and notations

Concerning the input data, the following notations are used:

– p is the number of product types to be realized;

– $\mathbf{N}_i$ is the set of operations corresponding to product i, i=1,2,…p;

– $\mathbf{N}=\bigcup_{i=1}^{p}\mathbf{N}_i$ is set of operations of the whole family;

– $\mathbf{N_b}=\bigcap_{i=1}^{p}\mathbf{N}_i$ is the operations set contained by all the products of the family;

– **B** is the set of blocks to realize the operations from **N** and **R** is the set of **B**'s indices;

– Cb_r is the cost of block r, tb_r is its operating time and Cs_0 is the cost of setting a new station;

– m_0 is the maximal number of workstations and n_0 is the maximal number of blocks assigned to a station;

– Δt is the time interval in which the demand of product i is n_i, i=1,2,…p.

From information on Δt and n_i one can establish the imposed cycle times for each product, Tc_i. It is supposed that **B** contains only blocks having operating times smaller than the smallest cycle time of the products: $tb_r \le \min_{i=1,2,\ldots p}\{Tc_i\}$ for all $r \in \mathbf{R}$.

It is assumed that there are three types of technological constraints: 1) the *precedence constraints*; 2) the *inclusion constraints* and 3) the *exclusion constraints*. Their formalizations are described here below.

1) The precedence relation over each set $\mathbf{N}_i$ is represented by an acyclic digraph $G_i=(\mathbf{N}_i,Dor_i)$. Note that the precedence relation is here taken in the non strict sense: an arc $(j,k)\in\mathbf{N}_i\times\mathbf{N}_i$ belongs to the set Dor_i if *either* operation k must be executed *after* operation j, *or* the two operations are made *in parallel*.

2) The exclusion constraints for allocation of blocks at the same station can be represented by a family *Dbs* of subsets $D\subseteq\mathbf{R}$ such that blocks from set D are forbidden of being allocated at the same station all together. These subsets result from considering both feasible station structure and manufacturing incompatibility between operations.

3) The inclusion constraints for allocation of operations at the same station can be represented, for each product i, by families D_i^{os} of subsets of operations from $\mathbf{N}_i$, such that all operations of the same subset must be performed at the same station. One can note that each Dos_i is at most a partition over the set $\mathbf{N}_i$.

Remark: In general, there exist inclusion and exclusion constraints of assigning operations at the same block; these are taken into account while forming the block set **B**. For our problem, it is supposed that all the blocks of **B** already meet these constraints.

2.2. Aggregation of constraints

As all the products must be realized on the same line, using the same equipment, an initial phase in treating a reconfigurable line optimization is to aggregate the constraints concerning the individual products. In particular, the aggregation of the precedence constraints to obtain a single precedence graph for the whole family will influence also the other two types of constraints.

The *aggregated* (or *total*) *precedence graph* is obtained by merging together the sets of individual

precedence relations. The steps are:

– represent all graphs superposed; the multiple vertices in the same sense are merged;

– delete vertices (i,j) if there is also a path from i to j;

– identify the *circuits* due to contrary precedences between operations in different individual graphs; the nodes from these circuits correspond to operations that cannot be separated without violating the precedence constraints, therefore, such operations are merged together into the so called *macro-operations*;

– redraw the total acyclic graph, where the macro-operations are represented by ordinary nodes.

Consequently, all the operation sets change ($\mathbf{N}_i$, i=1,2…p, and the total set, $\mathbf{N}$).

Concerning the *inclusion constraints*, each element of each D_i^{os}, i=1,2,…p, containing only parts of a macro-operation is extended with the absent operations. Then these sets are united and the elements having non empty intersection are merged together.

Also, the *blocks* that can execute only partially the macro-operations are to be eliminated. Consequently, the sets of *exclusion constraints* (elements of *Dbs*) in which these blocks exist must be eliminated too. These two latter actions may also be viewed as part of the model reduction.

The way of aggregating the constraints will be illustrated on the example from Section 4.

3. The proposed solving method

3.1. Linear programming (LP) formulation

The cost optimization of a reconfigurable transfer line may be formally presented as a LP problem. The model below resembles essentially to that for the single product case [19][20], but here the individual constraints are aggregated for all products, the blocks work sequentially in each station and the precedence relation is non strict.

The following variables and additional parameters are introduced:

– binary decision variables x_{rk}, where x_{rk}=1 if block r is assigned to station k and x_{rk}=0 otherwise, k=1,…,m_0;

– $y \geq 0$ denotes the number of stations;

– a lower bound m^* of the number of the stations;

– for each bloc r, the interval $K(r)=[head(r),tail(r)]$, where $head(r)$ is the earliest station and $tail(r)$ is the latest station where block r can be assigned;

– family $\mathrm{F_s}=\{F_1,...F_v\}$ of pairs of blocks having common operations: $F_q=\{r,t\}$ such that $B_r \cap B_t \neq \varnothing$ for any $q \in V=\{1,\ldots,v\}$ – i.e., $\mathrm{F_s}$ is a subset of alternative blocks (only one block from each pair of $\mathrm{F_s}$ can be used in a decision);

– $\mathrm{F_0}=\mathbf{B} \setminus \bigcup_{q \in V} F_q$, i.e., $\mathrm{F_0}$ is the set of blocks that will surely appear in the solution;

– $w_{rt}=|B_r \cap D_t|$ and $W_t=\{r \in \mathbf{R} \mid w_{rt}>0\}$ for any block r and any $D_t \in Dos$, that is, W_t are the blocks able to execute the operations involved in subset D_t of aggregated inclusion constraints;

– $U_t=\{r \in \mathbf{R} \mid i_t \in B_r\}$, where i_t is a given operation from the set $D_t \in Dos$;

– for each block B_r, the set $M(r)$ of operations not belonging to B_r which directly precede the operations of B_r;

– for each block r, the set $H(r)=\{t \in \mathbf{R} \mid B_t \cap M(r) \neq \varnothing\}$, that is the set of blocks capable of performing the operations from $M(r)$;

– $\mathbf{H}=\{r \in \mathbf{R} \mid M(r) \neq \varnothing\}$, i.e., the set of operations having predecessors;

– $h_{tr}=|B_t \cap M(r)|$ for any $r \in \mathbf{H}$ and any $t \in H(r)$;

– an upper bound R^* of the set of blocks to be assigned to the last station of the line.

The *objective function* corresponds to the line cost minimization:

$$Cs_0 \cdot y + \sum_{r=1}^{|\mathbf{R}|} \sum_{k=1}^{m_0} Cb_r \cdot x_{rk} \rightarrow \min \tag{1}$$

To speed up the computation, one may use also:

$$Cs_0 \cdot y + \sum_{r=1}^{|\mathbf{R}|} \sum_{k=1}^{m_0} (Cb_r + \varepsilon_r k) x_{rk} \rightarrow \min, \tag{1'}$$

where ε_r is a sufficiently small nonnegative value.

The following constraints ensures the execution of every operation from the aggregate operation set, $\mathbf{N}$, in exactly one station. Both cases are considered: when performing by blocks without intersection with the others (from $\mathrm{F_0}$), or by alternative blocks (from elements F_q of $\mathrm{F_s}$):

$$\sum_{k \in K(r)} x_{rk} \leq 1, \quad r \in \mathrm{F_0} \tag{2}$$

$$\sum_{r \in F_q} \sum_{k \in K(r)} x_{rk} \leq 1, \quad q \in V \tag{3}$$

All the operations from set $\mathbf{N}$ must be executed, so:

$$\sum_{r \in \mathbf{R}} \sum_{k \in K(r)} |B_r \cap \mathbf{N}| \cdot x_{rk} = |\mathbf{N}| \tag{4}$$

The aggregate precedence constraints on set $\mathbf{N}$ impose that:

$$\sum_{t \in H(r)} \sum_{s \in K(t), s \leq k} h_{tr} \cdot x_{ts} \geq |M(r)| \cdot x_{rk}, \quad r \in \mathbf{H}, \quad k \in K(r) \tag{5}$$

The aggregate inclusion constraints for the stations are met if:

$$\sum_{r \in W_t} w_{rt} \cdot x_{rk} = |D_t| \cdot \sum_{s \in U_t} x_{sk}, \quad D_t \in Dos, \quad k \in \bigcup_{s \in U_t} K(s) \tag{6}$$

Respect of the exclusion constraints for blocks at the same station writes as:

$$\sum_{r \in D_t} x_{rk} \leq |D_t| - 1, \quad D_t \in Dbs, \quad k \in \bigcap_{s \in D_t} K(s) \tag{7}$$

As only n_0 blocks per workstation are authorized, then:

$$\sum_{r \in \{t \in \mathbf{R} \mid k \in K(t)\}} x_{rk} \leq n_0, \quad k=1,2\ldots m_0 \tag{8}$$

The constraints concerning the number of stations require that:

$$y \geq k \cdot x_{rk}, \quad r \in R^*, \quad k \in K(r), \quad k \geq m^* \tag{9}$$

Finally, the cycle time constraints impose that:

$$\sum_{r \in \{t \in \mathbf{R} \mid k \in K(t)\}} t_r \cdot x_{rk} \leq Tc_i, \quad i=1,2,\ldots p, \quad k \geq m^* \tag{10}$$

In this model, the number of stations, y, depends on

the variables x_{rk}. The model does not explicitly contain the integrality constraint on y, but constraint (9) and the objective function (1) implicitly force it.

Next, some possible model reductions are proposed, in order to minimize the number of decision variables.

3.2. Model reduction

We propose here a short analysis of the block set after performing the aggregation of constraints and elimination of blocks able to execute only parts of macro-operations.

The first action is to examine if situations like the one described in Figure 1a) take place. In this figure, the precedence relations between two operations from different blocks are such as a "block circuit" appears. It is obvious that a solution cannot contain all the blocks from such a group; it is however sufficient that a single block be deleted. As the goal is the cost minimization, it is proposed that the most expensive as cost per operation block be eliminated. Note that such eliminations must start from the maximal circuits identified.

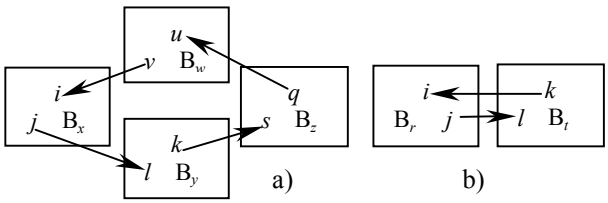


Figure 1. Example of blocks forming circuits and loops.

In particular, if two blocks r and t form a loop (like in Figure 1b)), it is not necessary that one of them be deleted, but certainly only one will appear in a solution. Therefore, it is sufficient that they be treated like the alternative blocks (i.e., the pair (r,t) be an element of F_s).

The second action to take in order to reduce the model is the following. For each block B_r from the last block set, a subset B' is searched, such that:

– operations from B' give the total set, **N**;

– $|B'| \le m_0 \cdot n_0$;

– all blocks from B' are mutually disjoint.

Each block for which such a subset B' does not exist must be eliminated.

Even if these reductions are not performed before the optimization phase, the optimizer will implicitly make them. But in large scale problems this could negatively affect the computation time.

3.3. Computation of bounds

We present here how the intervals $K(r)$ are computed for any block r, as well as the minimal number of stations, m^*, and the maximal block set to be assigned to the last station, R^*.

Intervals $K(r)$ and m^* are computed using the algorithm proposed in [15], based on the notion of *distance* between two operations. In our case, this distance can take only two values: 2, if the two operations can only be performed by blocks forbidden to be on the same station (i.e., belonging to elements of *Dbs*), and 0, otherwise.

In the cited work, the blocks are not a priori known. Therefore, first values $q^-(i)$ and $q^+(i)$ for any operation i must be computed, denoting the earliest and respectively the latest block where operation i can be assigned. To compute values $q^-(i)$, the algorithm needs as input data the total precedence graph, G, the aggregated inclusion constraints, *Dos*, and the distance matrix, d ($|\mathbf{N}|\times|\mathbf{N}|$). Values $q^+(i)$ result from the same algorithm, but entering the reversed precedence graph, G^r.

Then, values $k^-(i)$ and $k^+(i)$ of the earliest and the latest station where operation i can be assigned are computed, using the relation:

$k^{+/-}(i) = [q^{+/-}(i)/n_0]$,

where $[\cdot]$ denotes the smallest integer larger or equal with the argument.

Finally, for any block r, there are computed:

$$head(r) = \max\{k^-(i) |\ i \in B_r\} \quad (11)$$

$tail(r) = \min\{k^+(i) |\ i \in B_r\}$

The lower bound on the number of stations results as:

$$m^* = \max\{k^-(i) |\ i \in \mathbf{N}\} \quad (12)$$

Knowing intervals $K(r)$ for any block r, a value of R^* results from the following algorithm.

#1. $tail_max \leftarrow \max\{tail(r) |\ B_r \in \mathbf{B}\}$

#2. Find the minimum head of the blocks having the tail equal to *tail_max*. Let be *head_min* this minimum.

#3. Form the subset of blocks having the tail strictly larger than *head_min*. This subset is R^*.

The value of R^* can still be improved, which is aimed in the near future.

4. An illustrative example

A small scale example is presented to show how the proposed optimization procedure may be applied.

Let a product family be composed of p=3 products, given by their precedence graphs, as in Figure 2. The sets of operations are: $\mathbf{N}_1$={1,2,3,4,5,6,7,8,9,10}, $|\mathbf{N}_1|$=10; $\mathbf{N}_2$={1,2,3,4,5,6,7,8,9,10,11,12}, $|\mathbf{N}_2|$=12; $\mathbf{N}_3$={1,2,5,7,8,9,10,11,12,13}, $|\mathbf{N}_3|$=10. Thus, the total operation set is:

$$\mathbf{N} = \bigcup_{i=1}^{p} \mathbf{N}_i = \{1,2,3,4,5,6,7,8,9,10,11,12,13\},\ |\mathbf{N}| = 13,$$

and there are 7 common operations for all products:

$$\mathbf{N_b} = \bigcap_{i=1}^{\mathbf{p}} \mathbf{N}_i = \{1,2,5,7,8,9,10\}$$

The inclusion constraints (defining which operations must be performed on the same workstation) for product i, D_i^{os}, are represented on Figure 2 by ellipses. Thus:

$$D_1^{os} = \left\{ \underbrace{\{7,8\}}_{D_{11}^{os}}, \underbrace{\{9,10\}}_{D_{12}^{os}} \right\}, \qquad D_2^{os} = \left\{ \underbrace{\{7,8\}}_{D_{21}^{os}}, \underbrace{\{9,10,12\}}_{D_{22}^{os}} \right\} \quad \text{and}$$

$D_3^{os}=\left\{\underbrace{\{7,8\}}_{D_{31}^{os}},\underbrace{\{9,10\}}_{D_{32}^{os}}\right\}$. A supplementary set of inclusion constraints is $D_5^{os}=\{\{3,13\}\}$, concerning two operations belonging to different products, which have to be together when the products are realized on the same line (suggested by dotted rectangle in Figure 2). Taking into account the aggregation of constraints, this set will be treated just like any other D_i^{os}.

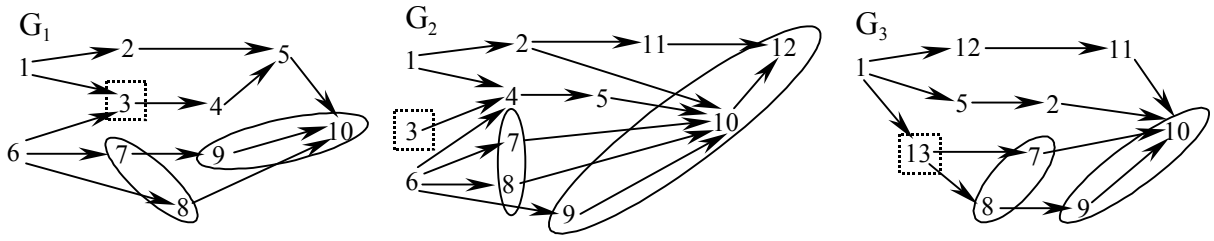


Figure 2. Precedence graphs and the inclusion constraints for a family of 3 products.

There is a unique set of exclusion constraints for all products, as they concern the (unique) block set. Thus, the sets of blocks forbidden to be assigned to the same station are:

$$D_{bs}=\left\{\underbrace{\{B_1,B_6,B_7\}}_{D_1^{bs}},\underbrace{\{B_1,B_9\}}_{D_2^{bs}},\underbrace{\{B_5,B_{11}\}}_{D_3^{bs}}\right\}$$

The total set **N** is to be executed by a set **B** of 12 multifunctional tools (blocks), whose features are provided in Table 1 (abbreviations t.u. and m.u. have been used for time units and monetary units respectively). The fixed cost of setting a station is Cs_0=350 m.u., the maximal number of stations is m_0=5 and the maximal number of blocks per station is n_0=3.

Block r	Operations	Operating time, tb_r [t.u.]	Cost, Cb_r [m.u.]
B_1	{1,3,6,13}	9	250
B_2	{1,3,13}	8	170
B_3	{1,2,7,8}	6	281
B_4	{2,5,9}	10	150
B_5	{2,5,7,8,11}	9	275
B_6	{2,6,9,10}	11	230
B_7	{4,6,8,10}	13	211
B_8	{4,7,8}	9	160
B_9	{4,7,8,9}	10	215
B_{10}	{5,12,13}	6	158
B_{11}	{10,11,12,13}	12	230
B_{12}	{2,5,10,11,12}	11	260

Table 1. Set of multifunctional tools (blocks) to realize the product family described in Figure 2.

It is required that in a period of Δt=48300 t.u. the following piece numbers of each product be made: n_1=2100, n_2=1932, n_3=2300. By using the expression $Tc_i=\Delta t/n_i$ for computing the required cycle time for product i, one obtains Tc_1=23 t.u., Tc_2=25 t.u. and Tc_3=21 t.u.

4.1. Preprocessing

The solving procedure begins by the *pre-processing phase* that allows also the model reduction. In this phase, the constraints aggregation will be performed first, leading to a single set of each type of constraints (as presented in §2.2), distinguished by the exponent "*pp*". Then, the model reduction is made on the "pre-processed" constraint sets.

Hereafter are shown the pre-processing steps on each type of constraints.

a) The phase of *generating the total precedence graph* as union of the three precedence graphs allows to identifying two circuits: 2→5→2 and 11→10→12→11. Therefore, two macro-operations are formed, denoted by a={2,5} and b={10,11,12}. The total precedence graph, G, defined on the new set of operations, $\mathbf{N}^{pp}$={1,3,4,*a*,*b*,5,6,7,8,13}, is depicted in Figure 3.

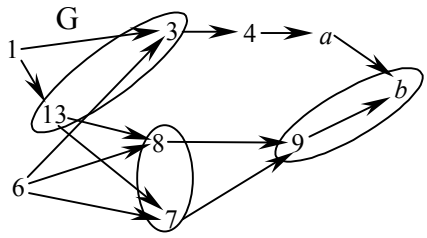


Figure 3. Precedence graph of the product family.

Because of the macro-operations, the operation set for each product has changed: $\mathbf{N}^{pp1}$={1,*a*,3,4,6,7,8,9,*b*}, $|\mathbf{N}^{pp1}|$=9; $\mathbf{N}^{pp2}$={1,*a*,3,4,6,7,8,9,*b*}, $|\mathbf{N}^{pp2}|$=9; $\mathbf{N}^{pp3}$={1,*a*,7,8,9,*b*,13}, $|\mathbf{N}^{pp3}|$=7.

b) *On the block set*, **B**, the pre-processing phase induces many types of changes.

b.1) The existence of *macro-operations* makes that those blocks which cannot execute all the operations from a macro-operation must be eliminated. In this case, these are B_3, B_5, B_6, B_7 and B_{10}. So, the new set

of blocks is $\mathbf{B}^{pp}=\{B_1,B_2,B_4,B_8,B_9,B_{11},B_{12}\}$, having noted that now $B_4=\{a,9\}$, $B_{11}=\{b,13\}$ and $B_{12}=\{a,b\}$.

b.2) The existence of "*circuits*" on a subset of $\mathbf{B}^{pp}$ must be verified (as shown in §3.2). In Figure 4 the thick arrows denote a precedence relation between two blocks (which exists *iff* all the operations from a block precede all the operations from the other block).

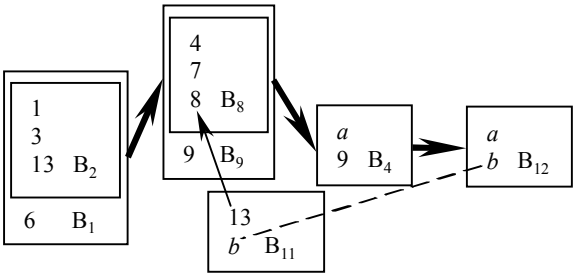


Figure 4. Circuits on the set of blocks, $\mathbf{B}^{pp}$.

The thin arrows represent precedence relation between operations. One can remark the existence of a circuit of blocks, that is $B_8 \rightarrow B_4 \rightarrow B_{12} \rightarrow B_{11} \rightarrow B_8$, from which the most expensive block as cost per operation must be eliminated. As from Table 1, this one is B_{11} (57.5 m.u. per operation). So, the new block set is:

$\mathbf{B}^{pp}=\{B_1,B_2,B_4,B_8,B_9,B_{12}\}$,

which still contains all the operations of $\mathbf{N}^{pp}$ (if this were not be the case, then the problem would not have any solution).

b.3) In small scale problems, this reduction is not very important. However, for large scale problems, it may be useful to make it in the pre-processing phase. In the analyzed case, blocks B_2, B_4 and B_8 may be eliminated.

c) Concerning the *inclusion constraints*, the three sets corresponding to products must be aggregated, taking into account the existence of macro-operations. Thus, each element of each D_i^{os}, i=1,2,3, and each element of D_s^{os} containing only a part of the operations included in macro-operations is extended with the absent operations: $D_1^{os\,pp}=\left\{\underbrace{\{7,8\}}_{D_{11}^{os\,pp}},\underbrace{\{9,\underbrace{10,11,12}_{b}\}}_{D_{12}^{os\,pp}}\right\}$,

$D_2^{os\,pp}=\left\{\underbrace{\{7,8\}}_{D_{21}^{os\,pp}},\underbrace{\{9,\underbrace{10,11,12}_{b}\}}_{D_{22}^{os\,pp}}\right\}$, $D_3^{os\,pp}=\left\{\underbrace{\{7,8\}}_{D_{31}^{os\,pp}},\underbrace{\{9,\underbrace{10,11,12}_{b}\}}_{D_{32}^{os\,pp}}\right\}$

and $D_s^{os\,pp}=\left\{\underbrace{\{3,13\}}_{D_{s1}^{os\,pp}}\right\}$. Then the union of these sets is formed and the elements having non empty intersection are merged together. Finally, the set of inclusion constraints is:

$$Dos^{pp}=\left\{\underbrace{\{3,13\}}_{D_1^{os\,pp}},\underbrace{\{7,8\}}_{D_2^{os\,pp}},\underbrace{\{9,b\}}_{D_3^{os\,pp}}\right\}$$

d) Changes on the set of *exclusion constraints*, *Dbs*, come from the elimination of some blocks in the previous pre-processing steps. Each element of Dbs means that the blocks cannot be *all together* on the same station (that is, the global exclusion relation does not mean mutually exclusion). Hence, if a block happens to disappear, then all the exclusion constraints containing it will also be eliminated. In our case, D_1^{bs} and D_3^{bs} are eliminated and the exclusion constraints after the pre-processing is:

$$Dbs^{pp}=\left\{\underbrace{\{B1,B9\}}_{D_1^{bs\,pp}}\right\}$$

4.2. Bounds

Once the pre-processing is finished, one must compute the bounds required by the linear solver. Following the steps described in §3.3, these are given in Table 2.

Block	*head*	*tail*	m^*	R^*
B_1	1	4	2	{1,2,4,8,9,12}
B_2	1	5		
B_4	2	5		
B_8	2	5		
B_9	2	5		
B_{12}	2	5		

Table 2. The bounds for the pre-processed problem of a three product family.

4.3. Optimization

According to §3.1, some elements need to be computed before the optimization phase. For the considered case, these are presented below and in Table 3.

$F_s=\{\{B_1,B_2\},\{B_4,B_{12}\},\{B_4,B_9\},\{B_8,B_9\}\}$

$F_0=\varnothing$

$\mathbf{R}=\{1,2,4,8,9,12\}$

$|Dos^{pp}|=n_{ic}=3$

$\mathbf{H}=\{B_4,B_8,B_9,B_{12}\}$

The optimal solution of the above described problem, as reported by the ILOG Cplex optimizer, is presented in Figure 5. It has the cost 1425 m.u. and consists of two workstations. The respect of the inclusion constraints is suggested by ellipses. One can note that, because of the exclusion constraints, blocks B_1 and B_9 have had to be assigned to different stations. Also, the cycle time has been met for all the three products.

t / r	w_{rt}, $r \in \mathbf{R}$, t=1,2,…n_{ic}=3			$M(r)$	$H(r)$	h_{tr}, $r \in \mathbf{R}$, $t \in \mathbf{R}$					
	W_1={B_1,B_2}	W_2={B_8,B_9}	W_3={B_4,B_{12}}			B_1	B_2	B_4	B_8	B_9	B_{12}
B_1	2	0	0	∅	∅	0	0	0	0	0	0
B_2	2	0	0	∅	∅	0	0	0	0	0	0
B_4	0	0	1	{4,7,8}	{B_8,B_9}	0	0	0	3	3	0
B_8	0	2	0	{3,6,13}	{B_1,B_2}	3	2	0	0	0	0
B_9	0	2	0	{3,6,13}	{B_1,B_2}	3	2	0	0	0	0
B_{12}	0	0	1	{4}	{B_8,B_9}	0	0	0	1	1	0

Table 3. Elements needed to instantiate the LP problem in the case of a three product family.

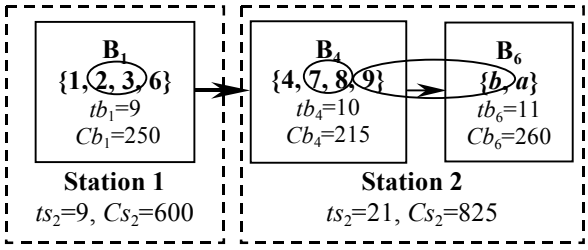


Figure 5. Optimal cost configuration of the transfer line for the three product family.

5. Concluding remarks. Future work

The problem of optimizing the cost of reconfigurable transfer lines aimed at producing a family of products is investigated. The transfer lines approached here are characterized by the possibility of parallelizing the operations' execution by the same spindle head. It is known that the initial capital commitment is the most important in the case of the transfer lines. The search of a unique optimal design decision for the whole family is thus justified by the potential economic benefit, possible from the very beginning. On the other hand, searching for the optimal solution may be time-consuming.

The cost optimization of the presented kind of transfer lines is a new and poorly studied problem, different from the classical ALB problem, but also NP-hard. This paper presented a linear programming approach to search an exact solution. An important phase of the solving procedure is the aggregation of constraints, which practically allows that the studied problem be treated like the single product one. Some proposals of model reduction have also been presented.

Next steps concern the improving of bounds, as well as numerical tests on a larger set of benchmarks. Due to the exponential complexity of the linear programming solving algorithm, it is expected a bad behavior when increasing the problem's dimension. The large number of constraints is a feature that will potentially allow the coupling of the presented exact method with different types of heuristics, which can provide good bounds to exact methods. This coupling is definitely necessary for applying the proposed method in real life environments.

Acknowledgments

Antoneta Iuliana BRATCU was supported by the Agence Universitaire de la Francophonie (A.U.F.) during the stage that she had at the École des Mines de Saint Étienne between October 2004 and May 2005.

References

[1] M. P. Groover, *Automation, production systems and computer integrated manufacturing*, Prentice Hall, Englewood Cliffs, New Jersey, 1987.

[2] K. Hitomi, *Manufacturing system engineering*, Taylor &Francis, 1996.

[3] G. Hutchinson, "Production Capacity: CAM vs. transfer line", *IE*, September 1976, pp. 30-35.

[4] G. W. Zhang, S. C. Zhang and Y. S. Xu, "Research on flexible transfer line schematic design using hierarchical process planning", *Journal of Materials Processing Technology*, Vol. 129, pp. 629-633, 2002.

[5] A. K. Kamrani and R. Logendran, *Group technology and cellular manufacturing: methodologies and applications*, Gordon and Breach, 1998.

[6] A. Scholl and R. Klein, "Balancing assembly lines effectively: a computational comparison", *European Journal of Operational Research*, Vol. 114, pp. 51-60, 1998.

[7] I. Baybars, "A survey of exact algorithms for the simple line balancing problem", *Management Science*, Vol. 32, pp. 909-932, 1986.

[8] F. B. Talbot, J. H. Paterson and W. V. Gehrlein, "A comparative evaluation of heuristic line balancing techniques", *Management Sciences*, Vol. 32, pp. 430-454, 1986.

[9] J. R. Johnson, "Optimally balancing large assembly lines with FABLE", *Management Science*, Vol. 34, 1988.

[10] E. Erel and C. Sarin, "A survey of the assembly line balancing procedures", *Production Planning and Control*, Vol. 9, pp. 414-434, 1998.

[11] B. Rekiek, A. Dolgui, A. Dechambre and A. Bratcu, "State of art of assembly lines design optimization", *Annual Reviews in Control,* Vol. 26/2, pp. 163-174, 2002.

[12] A. Scholl, *Balancing and sequencing assembly lines*, 2nd edition, Physica, Heidelberg, 1999.

[13] C. Becker and A. Scholl, "A survey on problems and methods in generalized assembly line balancing", *European Journal of Operational Research* (in press, available on line 11 September 2004).

[14] A. Dolgui, N. Guschinsky and G. Levin, "On problem of optimal design of transfer lines with parallel and sequential operations", *Proceedings of the 7th International Conference on Emerging Technologies and Factory Automation – ETFA'99,* (J.M. Fuertes Ed.), IEEE, Spain, pp. 329-334, 1999.

[15] A. Dolgui, N. Guschinsky and G. Levin, *Approaches for transfer lines balancing*, Preprint No. 8, Institute of Engineering Cybernetics/University of Technology of Troyes, 2000.

[16] A. Dolgui, N. Guschinsky and G. Levin, "Decomposition methods to optimize transfer lines with parallel and sequential machining", *Proceedings of the IFAC Conference on Management and Control of Production and Logistics: A Proceedings Volume of the Conference – MCPL 2000* (Grenoble, France), pp. 983-988, 2001.

[17] A. Dolgui, N. Guschinsky, G. Levin and Y. Harrath, "Optimal design of a class of transfer lines with blocks of parallel operations", *Proceedings of the IFAC Symposium on Manufacturing, Modeling, Management and Control – MIM 2000* (Patras, Greece), pp. 36-41, 2001.

[18] A. Dolgui, N. Guschinsky, G. Levin and Y. Harrath, "Un modèle de programmation linéaire pour l'équilibrage des charges des lignes de transfert" *European Journal of Automated Systems*, Vol. 36, pp. 4-31, 2001.

[19] A. Dolgui, N. Guschinsky, G. Levin, M. Louly, and S. Belmokhtar, "Balancing of Transfer Lines with Simultaneously Activated Spindles", *Preprints of the IFAC Symposium on Information Control Problems in Manufacturing – INCOM'04* (*CD-ROM*), April 5-7 2004, Salvador da Bahia, Brazil (to appear also in the *IFAC Proceedings Volume*, Elsevier).

[20] S. Belmokhtar, A. Dolgui, N. Guschinsky, I. Ihnatsenka and G. Levin, "Optimization of transfer line by constraint programming approach", *Proceedings of Computer and Industrial Engineering Conference* (*CD-ROM*), November 13-16 2004, San Francisco, U.S.A.

[21] A. Dolgui, N. Guschinsky and G. Levin, "A special case of transfer lines balancing by graph approach", *European Journal of Operational Research* (in press, available on line 11 September 2004).