5

Modular Machining Line Design and Reconfiguration: Some Optimization Methods

S. Belmokhtar, A.I. Bratcu and A. Dolgui

1. Introduction

1.1 Machining lines

Automated flow-oriented machining lines are typically encountered in the mechanical industry (Groover, 1987; Hitomi, 1996; Dashchenko, 2003). They are also called transfer (or paced) lines, being preferred mainly for the mass production, as they increase the production rate and minimize the cost of machining parts (Hutchinson, 1976). They consist of 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 spindle heads, each of these latter being composed of several tools. Each tool executes one or several (for the case of a combined cutting tool) operations. A *block* of operations is defined by the set of the operations executed simultaneously by one spindle head. When all blocks of a workstation have been accomplished, the workstation cycle time is terminated. The cycle time of the line is the longest workstation cycle time; its inverse is the line's production rate.

A machining line designed to produce a single product type is called *dedicated* line; its optimal structure, once found and implemented, is intended for a long exploitation time and needs high investments. The main drawback of such a system is its rigid structure which does not permit any conversion in case of change in product type. Thus, to react to changes effectively an alternative is to design the system from the outset 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* (Zhang *et al.*, 2002). This is the most important aspect which characterizes the potential of a system for reconfiguration (Koren *et al.*, 1999). The chapter deals with the designing of modular *reconfigurable* transfer lines, where a set of standard spindle heads are used to

Source: Manufacturing the Future, Concepts - Technologies - Visions , ISBN 3-86611-198-3, pp. 908, ARS/plV, Germany, July 2006, Edited by: Kordic, V.; Lazinica, A. & Merdan, M.

produce a *family of similar products*. The objective is to minimise the total investment cost and implicitly minimise the time for reconfiguration. For simplicity, in this chapter, "spindle heads" and "blocks" will have here the same meaning.

Our interest focuses on the configuration/reconfiguration of modular lines. The modularity brought many advantageous: maintenance and overhaul become easier, installation is rapid and reconfiguration becomes possible (Mehrabi et al., 1999). An approach to solve the problem is provided. Such approach is not limited to any specific system. It could be either used to configure a line for one time in case of DML (since the configuration is locked for the whole life time of the system) or to reconfigure the system in case of RMS at each time the demand changes to adapt to the new situation.

The design or configuration of a modular machining lines deals with the selection of modules from a given set and with their assignment to a set of stations. The modules in such lines are the multi-spindle units. Figure 1 illustrates a unit with 2 spindles. When the line has to be configured for the first time, i.e., the line has to be built, the given set of modules is formed on the basis of the following information:

a) The availability on the market, proposed by the manufacturers of spindle units.
b) The knowledge of the engineering team to design and manufacture their own spindle units
c) The already used spindle units which worked on the old lines and are still operational

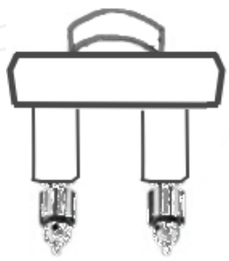

Figure 1. Two-spindle unit

The problem remains in finding an assignment for spindle units such that all operations are performed and all technological within cycle time constraints

are fulfilled. The objective is to minimize the total cost considering the cost of workstations and the costs of spindle units. Depending on the type of system we deal with, the costs can be either the fixed costs in case of dedicated lines or reconfiguration costs considering the amortization of the equipment.

A diagram of the design approach using our IP models is presented in Figure 2. This could be integrated in a holistic approach for line design which is similar to the framework of modular line design suggested by (Zhang *et al.*, 2002).

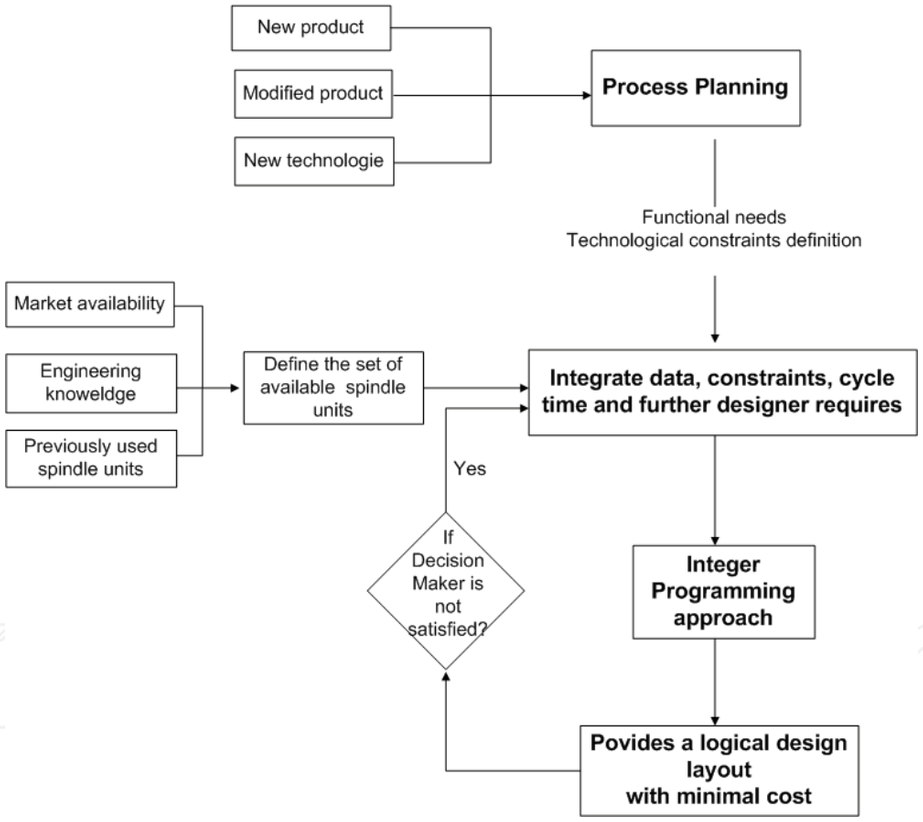

Figure 2. A global conceptual schema

In the next section related works the problem of designing modular machining lines for single product case is firstly addressed. Then, the proposed approach is generalized to the broader case considering a family of products.

1.2 Configuration and reconfiguration – related works

Koren *et al.* (1999) perform a comprehensive analysis of different types of manufacturing systems. Despite of their high cost, the dedicated machining lines (DML) are very effective as long as the demand exceeds the supply. Even these lines could operate at their full capacity; their average utilisation rate does not exceed 53%, as shown in the cited work. Flexible manufacturing systems (FMS), on the other hand, are built with the maximal available flexibility. They are able to respond to market changes, but are very expensive due to the involved CNC technology. The same study shows that 66% of FMS are not exploiting their full flexibility. Consequently, capital lies idle and an important portion is wasted.

A new alternative to the latter systems is brought by the *reconfigurable manufacturing systems* (RMS). The RMS aims to compensate the disadvantages of the last systems. This can be achieved by combining the high productivity of DML and flexibility of FMS, hence, providing a cost-effective and quick response to market changes. The cited authors define the RMS as being "designed at the outset for rapid change in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements."

Youssef & ElMaraghy (2005) identify two aspects of the reconfiguration, namely the software part and the hardware (physical) part. The effort is placed in the first part – machine re-programming, re-planning, re-scheduling, re-routing – whereas the physical reconfiguration relies upon adding/removing machines and changing the handling material. Son (2000) proposes a genetic algorithm based design methodology of an economic reconfiguration in response to demand variations, by using a configuration similarity index.

RMS must necessarily be based on a *modular* structure to meet the requirements for changeability. To configure machining systems the interfaces between the modules have prior importance, therefore these latter must meet some standard specifications. A first step to reconfigurability and, meanwhile, its strongest justification, is to ensure the possibility of producing a family of products, instead of a single one, such that to enable a smooth re-adaptation of the system to a continuously changing demand. A low cost design of a mixed-model machining line will implicitly ensure the re-adaptation time minimisation also.

A single-part versus multiple-part manufacturing systems (SPMS vs. MSPS) critical analysis is performed by Tang *et al.* (2005a). The SPMS concern the production of a single type of product on a practically rigid line configuration, whereas the MSPS are intended for a product family. The MSPS are obviously more complex, need a more sophisticated transfer system, but conversely offer more flexibility at a lower cost.

In the following, we give a formal description of the designing machining line problem for single product.

1.3 Single product case

1.3.1 Problem description

In order to model the problem we have to understand the mechanism of the machining process. We consider the lines where the activation of the spindle units at workstations is sequential. At the level of a workstation, the spindle units operate one after another on the positioned part to be manufactured. So, each workstation has an execution time equal to the sum total of its spindle times. The cycle time of the line is the elapsed time between the starting machining of the spindle units and their end on all workstations. Thus, the cycle time is determined by the slowest station of the line. At the end of each cycle time, the parts are moved to the next station and another cycle begins. Figure 3 illustrates such a line with 2 workstations. The first workstation is equipped with 2 multi-spindles whereas the second has only one unit. Each unit is composed of two spindles.

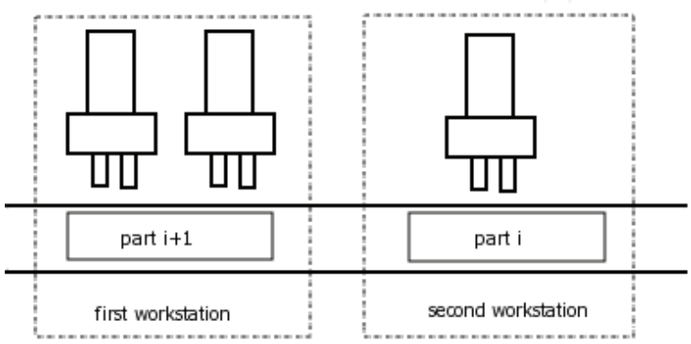

Figure 3. An example of a dedicated line

Notations and assumptions are as follows:

- **N** is the set of operations that have to be performed on each part (drilling, milling, boring, etc.).
- **B** = $\{B_r | B_r \subset \mathbf{N}\}$ corresponds to the set of all available multi-spindle units, where each is defined by the subset of operations it performs. A multi-spindle unit B_r is physically composed of one or several tools performing simultaneously corresponding operations. For the sake of simplicity, the term *block* is used henceforth to refer to a multi-spindle unit. Thus, the block $B_r \subset \mathbf{N}$ is said to contain the operations performed by the corresponding multi-spindle unit.
- q_r is the cost of block B_r,
- t_r is the execution time of block B_r,
- C_s is the average cost for any created workstation,
- C_T is the cycle time to not exceed,
- m_0 and n_0 are the maximum number of workstations which can be created and the maximum number of blocks which can be assigned to any workstation, respectively.

It is assumed that the following constraints are known:

1. cycle time,
2. precedence relations between operations,
3. exclusion conditions for blocks and
4. inclusion constraints for operations to be executed at the same workstation.

The above constraints are defined as follows:

1. An upper bound on the cycle time insures a minimal threshold of throughput.
2. Precedence relations impose an order which should be respected between some operations. For example, before drilling or boring one part a jig boring should be performed. A jig boring consists in making a notch when "true position" locating is required. The order relation over the set **N** can be represented by an acyclic digraph $G^{or} = (\mathbf{N}, D^{or})$. An arc $(i, j) \in \mathbf{N} \times \mathbf{N}$ belongs to the set D^{or} if the operation j must be executed after operation i.

3. Exclusion conditions correspond to the incompatibility between some operations, e.g. it can be the inability to activate some tools on the same workstation. The same kind of constraints have been already studied by Park, Park and Kim (1996) where the assignment of tasks may be restricted by some incompatibilities (minimum or maximum distances in terms of time or space between stations performing a pair of tasks). In our case, this incompatibility is extended to blocks such that blocks involving incompatible operations are not assigned to the same workstation. The constraints are represented by a collection D^{bs} of subsets $D_l \subseteq$ B such that all blocks from the set D_l cannot be assigned to the same workstation. But any subset strictly included in D_l can be assigned to the same workstation.

4. Restrictions related to operations which have to be executed on the same station are referred to as inclusion relations. For example, if a precise distance is required between two holes, the operations corresponding to their drilling should be performed at the same workstation. If these operations are performed on different workstations, then the impact of moving reduces greatly the chance of successful precision drilling for subsequent holes. The inclusion conditions can be represented by a family D^{os} of subsets $D_t \subseteq$ N such that all operations of the same subset D_t from D^{os} must be performed at the same workstation. In Pastor and Corominas (2000) similar restrictions are considered, these operations are introduced as one operation. Beyond the possibility of merging the operations, we also consider the case where operations can be performed separately with different spindle units (if such units are available).

1.3.2 The integer linear program

Decision variables are defined as follows:

$$x_{rk} = \begin{cases} 1, & \text{if block } B_r \text{ is assigned to station } k \\ 0, & \textit{otherwise} \end{cases}$$

$$y_k = \begin{cases} 1, & \text{if station } k \text{ is opened} \\ 0, & \textit{otherwise} \end{cases}$$

Additional parameters have to be defined, they are described as follows.

- $K(r) = [head_r, tail_r] \subseteq [1, m_0]$ is the interval of the workstation indices where block $B_r \in \mathbf{B}$ can be assigned. The $head_r$ is the earliest station where block B_r can be assigned and $tail_r$ is the last; $head_r$ and $tail_r$ values are computed on the basis of problem constraints. Obviously, the number of decision variables is directly proportional to the width of the interval $K(r)$;
- $Q(i) = \{B_r \in \mathbf{B} \mid i \in B_r\}$. Thus, $Q(i)$ contains all blocks from **B** which perform operation $i \in \mathbf{N}$;
- interval $KO(j)$ corresponds to all stations where operation j can be performed:

$$KO(j) = \bigcup_{B_r \in Q(j)} K(r)$$

The objective function is expressed as follows:

$$\text{Minimize} \sum_{k=m^*+1}^{m_0} C_s \cdot y_{k+} \sum_{k=1}^{m_0} \sum_{B_r \in \mathbf{B}} q_r \cdot x_{rk}$$

The following constraints ensure for each operation from set **N** its execution in only one workstation:

$$\sum_{B_r \in Q(i)} \sum_{k \in K(r)} x_{rk} = 1, \quad \forall i \in N$$

The cycle time constraints for each workstation are:

$$\sum_{B_r \in \mathbf{B}} x_{rk} \cdot t_r \le C_T, \quad \forall k = 1, 2, \ldots, m_0$$

The precedence constraints must not be violated:

$$\sum_{B_r \in Q(i)} \sum_{l=head_r}^{k-1} x_{rl} \ge \sum_{B_r \in Q(j)} x_{rk}, \quad \forall (i,j) \in D^{or}, \quad \forall k \in KO(j)$$

The inclusion constraints for operations are respected with the following constraints:

$$\sum_{B_r \in Q(i)} x_{rk} = \sum_{B_r \in Q(j)} x_{rk}, \quad \forall \{i,j\} \subseteq D_t, \quad \forall D_t \in D^{os}, \quad \forall k \in KO(j)$$

The exclusion constraints for blocks are respected if:

$$\sum_{\mathrm{B}_r \in D_l} x_{rk} \le |D_l| - 1, \quad \forall D_l \in D^{bs}, \quad \forall k \in \bigcap_{\mathrm{B}_r \in D_l} K(r)$$

The maximal number of blocks by workstation is respected by:

$$\sum_{\mathrm{B}_r \in \{\mathrm{B}_s \in \mathbf{B} | k \in K(s)\}} x_{rk} \le n_0, \quad \forall k = 1,2,...,m_0$$

The following constraints are added in order to avoid the existence of intermediate empty workstations:

$$y_{k-1} - y_k \ge 0, \quad k = m^* + 2,....,m_0$$

$$y_k \ge x_{rk}, \quad k \ge m^* + 1, \quad B_r \in \{B_s \in \mathbf{B} | k \in K(s)\}$$

In the next section we show how the model for a single product, above presented, can be extended to a family of products. Many of the assumptions and notation are maintained, and some new assumptions have to be considered, as shown below.

1.4. Family product case

1.4.1 Problem description

The features of the product family are supposed known, each product being described by the corresponding precedence graph and the required cycle time. An admissible resemblance degree must be assumed between products, as to some well known rules of defining a product family (Kamrani & Logendran, 1998) (for example, they must have a minimal number of common operations).

The goal is to design the minimal cost line configuration from the given set of available modules. This configuration must ensure a desired throughput level. By designing, we mean to determine the number of workstation to establish and to equip them with blocks such that all operations are executed only once. We are interested in the best structure of such line with regard to fixed cost point of view. Thus, the objective function is a linear combination of the cost of workstations to be established and the costs of blocks chosen to be assigned to them.

1.4.2 Problem assumptions

For our problem 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;
- the activation of blocks belonging to the same station is sequential;
- station setting cost, blocks' operating times and cost of each block are given.

An admissible assignment of blocks to stations is to find such that all the technological constraints – order, compatibility and desired cycle times – be satisfied and the *total equipment cost* (for all stations and blocks) is minimal. We are not interested to find the blocks activation sequence for each product, but we should ensure for the provided solution that such an order always exists for each product.

1.4.3 Related literature

This kind of problems is known in literature as *Line Balancing* problems (Scholl & Klein, 1998). In case of a single product type and if each block is composed of only one operation, then the problem is reduced to the basic problem, *Simple Assembly Line Balancing* (SALB). The aim 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.

In general, integer linear programming models for the SALBP are formulated and solved by exact or heuristic methods (Baybars, 1986; Talbot *et al.*, 1986; Johnson, 1988; Erel & Sarin, 1998; Rekiek *et al.*, 2002). The *Mixed-model Assembly Line Balancing* (MALB) problem approaches the optimization of lines with several "versions" of a commodity in an intermixed sequence (Scholl, 1999). The design of such lines – also called *multi-product* or *multi-part* lines – must take into account the possible differences between versions – among others, different precedence relations, different task times, etc. By enriching the basic assumptions of SALB, the *Generalized* ALB (GALB) problems have also been stated, in order to solve more realistic problems – a comprehensive survey may be found in Becker & Scholl (2006).

The balancing problems with *equipment selection* have some common features with the studied problem. In this context, a recent approach proposes the use of a genetic algorithm for configuring a multi-part optimal line, having the maximal efficiency (minimal ratio of cost to throughput) as criterion for the fitness function (Tang *et al.*, 2005b). Our problem differs essentially from the balancing and equipment selection problems because the operations are simultaneous into blocks. This feature makes it impossible to directly solve by the known methods.

The closest problems are studied in Dolgui *et al.* (1999), Dolgui *et al.* (2000), Dolgui *et al.* (2001), Dolgui *et al.* (2005) and Dolgui *et al.* (2006b) where all blocks 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 is a potential block). In Dolgui *et al.* (2004), Belmokhtar *et al.* (2004), Dolgui *et al.* (2006a) the blocks are known and are executed in parallel. All these papers concern the case of a single product, for which three solving approaches have been proposed: a constrained shortest path; mixed integer programming (MIP); heuristics. A generalization of the linear programming approach to the case of a product family and sequential activation of blocks was for the first time presented by Bratcu *et al.* (2005).

The rest of this chapter is organised as follows. In Section 2 a detailed formal description of the problem is presented, along with the needed notations and explanations on how the constraints' aggregation is made. In Section 3 the proposed solving procedure is discussed, based upon a linear programming model, possibly to improve by some reductions. Section 4 is dedicated to some concluding remarks and to perspectives.

2. Formal statement of the problem

2.1 Input data and notations

The problem is identified by answering to the following questions:

a) *what* must be produced? – this is the set of features characterizing the product family (number of products, set of operations and precedence constraints for each product);
b) *how* should them be produced? – these are the blocks' characteristics (cost and operating time of each block);

c) production conditions (*external* environment – like demand, for example – and *internal constraints* – for example, maximal number of stations or maximal number of stations on the line).

As consequence, the following input data are given for each instance of the problem:

a) - p is the number of product types to manufacture;
   - $\mathbf{N}_i$ 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;
b) - **B** is the set of blocks for realizing the operations from **N**, with **R** being the set of **B**'s indices;
   - Cb_r is the cost of block r and tb_r is its operating time;
c) - Cs0 is the cost of setting a new station;
   - m0 is the maximal number of workstations and n0 is the maximal num ber of blocks assigned to a station;
   - Δt is the time interval in which the demand of product i is ni, i=1,2,…p.

From quantitative information about the demand, that is, from Δt and n_i, the imposed cycle times for each product, Tc_i, may be computed. It is assumed that **B** contains only blocks having operating times smaller than the smallest cycle time of the products: $tb_r \leq \min_{i=1,2,\ldots p} \{Tc_i\}$ for all $r \in \mathbf{R}$.

Three types of technological constraints are considered:

1. the precedence constraints
2. the inclusion constraints
3. the exclusion constraints.

Their meanings and formalizations are detailed hereafter:

1. The precedence relation is a partial order relation over each set $\mathbf{N}_i$. It is represented by an acyclic digraph $G_i=(\mathbf{N}_i, Dor_i)$. One should notice that the precedence relation is here taken in the non strict sense: a vertex $(j,k) \in \mathbf{N}_i \times \mathbf{N}_i$ belongs to the set Dor_i if *either* operation k must be executed *after* operation j, *or* the two operations are performed *in parallel* (in the same time).

2. Exclusion conditions correspond to incompatibility between some operations and have the same meaning like in the single product case (see section 1.3.1).
3. Restrictions related to operations which have to be executed on the same station are referred to as inclusion relations. These also have the same meaning like in the single product case.

The inclusion conditions can be represented by a family D_i^{os} of subsets $D_t \subseteq \mathbf{N}_i$ such that all operations of the same subset D_t from D_i^{os} should be performed at the same workstation. One can note that each D_i^{os} is at most a partition over the set $\mathbf{N}_i$. *Remark*: In the general case, where the blocks of operations are not known, there exist inclusion and exclusion constraints of assigning operations to the same block, that is, sets of operations forbidden to be assigned to a block all together, respectively sets of operations which are mandatory to be assigned to a same block. For our problem, these constraints are taken into account while forming the block set **B**, therefore it is supposed that all the blocks of **B** already meet these constraints.

2.2 Aggregated constraints

As all the products should be produced on the same line, using the same equipment, an initial phase in dealing with a reconfigurable line optimization is the aggregation of the constraints concerning the individual products. Due to the assumption on the same characteristics for products belonging to the same family, the constraints should not be contradictory. In case where it happens, there will be no feasible solution to the design problem: a line for machining the given product family cannot be designed under the given constraints.

In particular, there should normally not be contradictory precedence constraints between operations common to several products of the family. But if this however happens, one must first make the aggregation of the precedence constraints to obtain a single precedence graph for the whole family. This operation will influence also the other two types of constraints, as shown later. The *aggregated* (or *total*) *precedence graph* is obtained by merging together the sets of individual precedence relations, according to the following steps:

- represent all graphs superposed and merge the multiple vertices in the same sense;
- delete redundant arc (i,j), i.e., if there is a path from i to j (containing several transitive vertices), then the arc (i,j) is said to be redundant and consequently should be deleted;
- identify the *circuits* due to contrary precedence 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 newly introduced *macro-operations*;
- redraw the total acyclic graph, where the macro-operations are represented by ordinary nodes.
- The definition of the macro-operations will consequently induce changes in all the operation sets, $\mathbf{N}_i$, i=1,2…p, and also in the total set, $\mathbf{N}$, as well as in the set of both inclusion and exclusion constraints.

Concerning the *inclusion constraints*, each element of each D_i^{os}, i=1,2,…p, containing only some operations of a macro-operation, is extended with the absent operations. These sets are then united and the elements having non empty intersection are merged together. Furthermore, the *blocks* which execute just a part of the macro-operations should be eliminated; the final, aggregated set of inclusion constraints is denoted by *Dos*. Next, the sets of *exclusion constraints* (elements of *Dbs*) containing these blocks are to be eliminated too. These two latter actions may also be viewed as part of the model reduction (detailed in Section 3.2).

2.3 Example

Here below is detailed an example to illustrate the aggregated constraints.

Let a product family be composed of p=3 products, given by their precedence graphs, as in Figure 4. The corresponding sets of operations are:

$$\mathbf{N}_1=\{1,2,3,4,5,6,7,8,9,10,11\},\ |\mathbf{N}_1|=11,$$
$$\mathbf{N}_2=\{1,2,3,4,5,6,7,8,9,10,11,12\},\ |\mathbf{N}_2|=12,$$
$$\mathbf{N}_3=\{1,2,4,5,7,8,9,10,11,12,13\},\ |\mathbf{N}_3|=11.$$

The total operation set is therefore:

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

and there are 9 common operations for all products:

$$\mathbf{N_b} = \bigcap_{i=1}^{\mathbf{p}} \mathbf{N}_i = \{1,2,4,5,7,8,9,10,11\}$$

Ellipses in Figure 4 represent the inclusion constraints (defining which operations must be performed on the same workstation) for product *i*, D_i^{os} . The corresponding sets are:

$$D_1^{os} = \left\{ \underbrace{\{7,8\}}_{D_{11}^{os}}, \underbrace{\{9,10\}}_{D_{12}^{os}} \right\},\ D_2^{os} = \left\{ \underbrace{\{7,8\}}_{D_{21}^{os}}, \underbrace{\{9,10,12\}}_{D_{22}^{os}} \right\} \text{ and } D_3^{os} = \left\{ \underbrace{\{7,8\}}_{D_{31}^{os}}, \underbrace{\{9,10\}}_{D_{32}^{os}} \right\};$$

to these ones a supplementary set of inclusion constraints is added, $D_s^{os} = \{\{3,13\}\}$, concerning two operations belonging to different products, which have to be together when the products are realized on the same line, This latter set is suggested by dotted rectangle in Figure 4. Taking into account the aggregation of constraints, this set will be treated just like any other D_i^{os} .

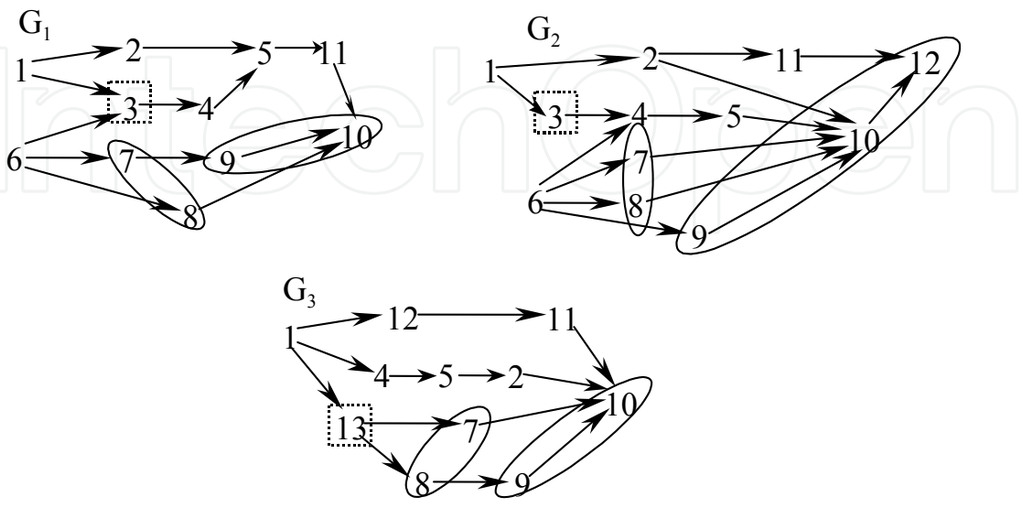

Figure 4. Precedence graphs and inclusion constraints for a family of 3 products.

Next, suppose that the total set **N** is to be executed by a set **B** of 12 multifunctional tools (blocks), whose features – operations to perform, operating times and costs – are provided in Table 1 (abbreviations t.u. and m.u. denote respectively time units and monetary units).

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 blocks to manufacture the product family described in Figure 4.

The fixed cost of setting a new 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.

Next, suppose that the following constraints related to minimal line throughput are imposed: in a period of Δt=48300 t.u. n_1=2100 pieces of the first product , n_2=1932 pieces of the second product and n_3=2300 pieces of the third product must be manufactured. Computing the required cycle with the expression $Tc_i=\Delta t/n_i$ for time for product i, one obtains Tc_1=23 t.u., Tc_2=25 t.u. and Tc_3=21 t.u. respectively.

The exclusion constraints are provided by a unique block set for all products. Suppose that the sets of blocks forbidden to be assigned to the same station are:

$$D^{bs}=\left\{\underbrace{\{B_1,B_6,B_7,B_{10}\}}_{D_1^{bs}},\underbrace{\{B_1,B_9\}}_{D_2^{bs}},\underbrace{\{B_3,B_5,B_{11}\}}_{D_3^{bs}}\right\}$$

The constraints aggregation is part of a pre-processing performed on the initial data about the problem; it will lead to a single set of each type of constraints, distinguished by the exponent "*pp*" (acronym corresponding to *p*re-*p*rocessing). The generation of the total precedence graph – a single one for the whole product family – by merging together the individual precedence graphs, is the starting point in aggregating constraints. In the considered case, this operation allows to identifying two circuits, 2→5→2 and 11→10→12→11, due to contrary precedence relations between same operations in different individual graphs. Therefore, two macro-operations are formed, denoted by *a*={2,5} and *b*={10,11,12}. The total precedence graph, G, defining a partial order relation on the new set of operations, $\mathbf{N}^{pp}$={1,3,4,*a*,*b*,5,6,7,8,13}, is shown in Figure 5.

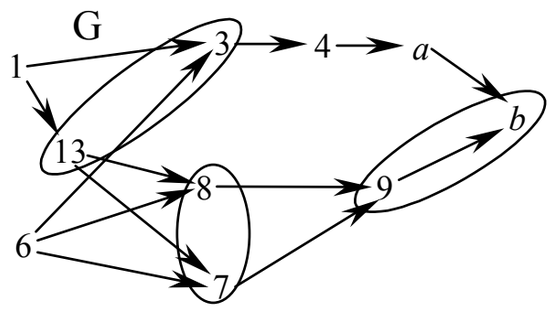

Figure 5. Precedence graph of the whole product family.

As the macro-operations have been introduced, the *operation set* for each product has also changed:

$$\mathbf{N}^{pp1}=\{1,a,3,4,6,7,8,9,b\},\ |\mathbf{N}^{pp1}|=9,$$
$$\mathbf{N}^{pp2}=\{1,a,3,4,6,7,8,9,b\},\ |\mathbf{N}^{pp2}|=9,$$
$$\mathbf{N}^{pp3}=\{1,a,4,7,8,9,b,13\},\ |\mathbf{N}^{pp3}|=8.$$

Also, those *blocks* which cannot execute but only some operations from a macro-operation must be eliminated. These are B_3, B_5, B_6, B_7 and B_{10}. Therefore, 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*}.

Next, one must perform the aggregation of the *inclusion constraints*, 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 some of the operations included in macro-operations is first extended with the absent operations:

$$D_1^{os\,pp} = \left\{ \underbrace{\{7,8\}}_{D_{11}^{os\,pp}}, \underbrace{\left\{ \underbrace{9,10,11,12}_{b} \right\}}_{D_{12}^{os\,pp}} \right\}$$

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

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

Set $D_s^{os\,pp} = \left\{ \underbrace{\{3,13\}}_{D_{s1}^{os\,pp}} \right\}$ remains unchanged. Then, all these 4 sets are united and the elements having non empty intersection are merged together. The aggregated set of inclusion constraints is:

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

The set of *exclusion constraints*, *Dbs*, must be changed because of the elimination of some blocks in the previous aggregation steps. Each element of *Dbs* has the meaning of forbidding the blocks to be *all together* on the same station (note that the global exclusion relation does not necessarily mean mutually exclu-

sion). Hence, if a block happens to be eliminated, then all the exclusion constraints containing it will also be eliminated. In the considered example, sets D_1^{bs} and D_3^{bs} are those to be eliminated. Therefore, the final exclusion constraints set is:

$$D^{bs\,pp} = \left\{ \underbrace{\{B_1, B_9\}}_{D_1^{bs\,pp}} \right\}$$

3. Solving by integer linear programming (IP)

3.1 IP formulation

The cost optimization of a reconfigurable machining line admits a IP formulation. The presented model is an extension of the one built for the single product case (Dolgui *et al.,* 2004; Belmokhtar *et al.*, 2004). The main difference is that individual constraints are aggregated for all products, the blocks work sequentially in each station and the precedence relation is not strict (see above).

The model needs that the following variables and additional parameters be introduced:

- binary decision variables x_{rk}, with x_{rk}=1 if block r is assigned to station k and x_{rk}=0 otherwise, k=1,…, m_0;
- $y \geq 0$ to denote the number of stations;
- m^* to denote a lower bound of the number of stations;
- for each block r, the interval $K(r)$=[$head(r)$,$tail(r)$], with $head(r)$ being the earliest station and $tail(r)$ being the latest station where block r can be assigned;
- family 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,…,v} – i.e., only one block from each pair of F_s can be used in a decision; F_s is called the subset of (pairs of) *alternative* blocks;
- F_0=$\mathbf{B} \setminus \bigcup_{q \in V} F_q$ – i.e., F_0 is the set of blocks that will surely appear in the solution;

- $w_{rt}=|B_r \cap D_t|$ and $W_t=\{r \in \mathbf{R} |\ w_{rt}>0\}$ for any block r and any $D_t \in Dos$, that is, W_t are the blocks able to execute the operations belonging to subset D_t of aggregated inclusion constraints;
- $U_t=\{r \in \mathbf{R} |\ i_t \in B_r\}$, where it 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 $H(r)=\{t \in \mathbf{R} |\ B_t \cap M(r) \neq \varnothing\}$, containing the blocks capable of performing the operations from $M(r)$;
- $\mathbf{H}=\{r \in \mathbf{R} |\ M(r) \neq \varnothing\}$, i.e., the set of operations having predecessors;
- $h_{tr}=|B_t \cap M(r)|$ for any $r \in \mathbf{H}$ and any $t \in H(r)$;
- R^* to denote an upper bound of the set of blocks to be assigned to the last station of the line.

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

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

which, for reasons of speeding up computation, can be also expressed as:

$$Cs_0 \cdot y + \sum_{r=1}^{|\mathbf{R}|} \sum_{k=1}^{m_0} (Cb_r + \varepsilon_r k) x_{rk} \rightarrow \min \qquad (1.1)$$

where ε_r is a sufficiently small nonnegative value. The optimization is subject to a set of constraints, whose mathematical forms are given and explained hereafter.

The first constraints ensures the execution of every operation from the aggregate operation set, **N**, in exactly one station. Both cases are considered: either choosing blocks without intersection with the others (from F_0), or choosing alternative blocks (from elements F_q of F_s):

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

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

As all the operations from the total set **N** must be executed, it holds that:

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

The aggregate precedence constraints on set **N** impose that:

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

The aggregate inclusion constraints for the stations are met if:

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

Respect of the aggregate exclusion constraints for assigning blocks to the same station writes as:

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

As n_0 is the maximal number of blocks to be allocated to a workstation, then:

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

The constraints concerning the number of stations require that:

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

The last constraints impose that the cycle time requirements be met:

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

In the above model one can note the dependence of the number of stations, y, on the variables x_{rk}. The model does not explicitly claim the integrality constraint on y, but constraint (9) and the objective function (1) implicitly force it.

Some possible model reductions may be performed, in order to minimize the number of decision variables, as proposed below.

3.2 Reduction of model and computation of bounds

In order to reduce computation time, an analysis of the block set after performing the aggregation of constraints – that is, after identifying the macro-operations – can allow some supplementary block eliminations. The steps presented hereafter are not mandatory, but can contribute to avoid useless computation.

The first action is to check if situations like the one described in Figure 6a) happen. In this figure, the precedence relations between two operations from different blocks are such that a "block circuit" appears. Obviously, a solution cannot contain all the blocks involved in such a circuit, but it is however sufficient that a single block be deleted. It is proposed that a heuristic elimination rule be used in this case, namely the most expensive – as cost per operation – block be eliminated, which is consistent with the goal of the total investment cost minimization. Note that such eliminations must start from the maximal circuits identified.

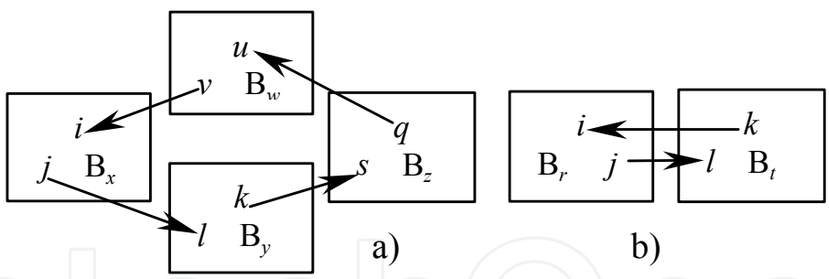

Figure 6. Example of blocks forming circuits and loops.

A special case is that of two vertices circuits (loops). If two blocks r and t form a loop (like in Figure 6b)), it is not necessary that one of them be deleted, but certainly only one will appear in a solution. It is therefore sufficient to treat them as alternative blocks (i.e., the pair (r,t) be an element of F_s).

The second step of the model reduction concerns also the set of blocks. Remember that this set, **B**, will have already undertaken some changes due to the constraints aggregation, as above mentioned.

Thus, for each block B_r from the last block set **B**, a subset B' is searched, such that:

- operations from B′ give the total set, **N**;
- $|B'| \leq m_0 \cdot n_0$;
- 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.

Hereafter are presented the reductions possible for the example considered in Section 2.3.

The first reduction step is to check the existence of "*circuits*" on subsets of $\mathbf{B}^{pp}$. It is said that a precedence relation exists between two blocks if and only if all the operations from a block precede all the operations from the other block. In Figure 7 precedence relations between two blocks have been represented by thick arrows, whereas the thin arrows denote precedence relation between operations.

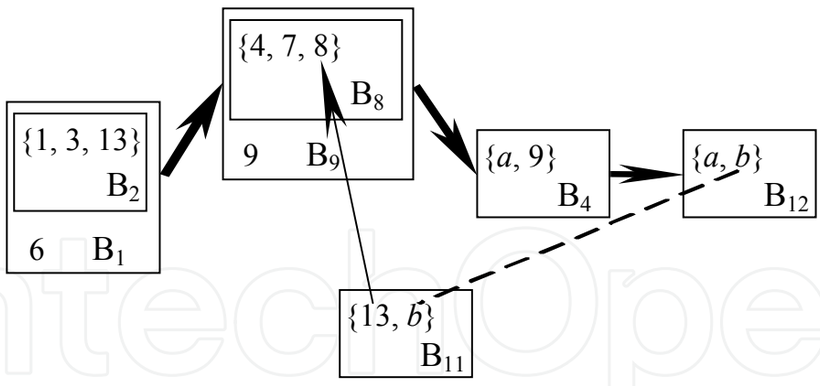

Figure 7. Circuits on the set of blocks after the constraints aggregation, B^{pp}.

One can remark the existence of a circuit of blocks, that is, $B_8 \rightarrow B_4 \rightarrow B_{12} \rightarrow B_{11} \rightarrow B_8$. According to the heuristic of eliminating the most expensive block as cost per operation, block B_{11} (57.5 m.u. per operation) must be eliminated, as to data provided in Table 1, Hence, the reduced 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).

Concerning the second reduction step, this is not very important in small scale problems. But for large scale problems, it may be useful to make it in the pre-processing phase. In the analyzed case, one can verify that blocks B_2, B_4 and B_8 may be eliminated.

As for the computation of bounds, we briefly present here below 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, R^*, to be assigned to the last station.

Intervals $K(r)$ and m^* are computed based upon the algorithm proposed by Dolgui *et al.* (2000), using the notion of *distance* between two operations. In the general case, this distance takes one of three values (0, 1 or 2) – in our case, it can take only two values: either 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*), or 0, otherwise.

In the cited work, the blocks are not a priori known. Therefore, the problem is solved in two steps: first determining the bounds of assigning operations to blocks, then for allocating blocks to workstations.

Thus, the algorithm begins with computing the values $q^-(i)$ and $q^+(i)$ for any operation i, which denote the earliest and respectively the latest block where operation i can be assigned. In our case, 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.

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

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

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

For any block r, there are finally computed:

$$\begin{cases} head(r) = \max\{k^-(i) \mid i \in B_r\} \\ tail(r) = \min\{k^+(i) \mid i \in B_r\} \end{cases} \quad (11)$$

The lower bound on the number of stations results as:

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

Having computed intervals $K(r)$ for any block r, a sufficiently good value of R^* may result from the following algorithm.

1. $tail\_max \leftarrow \max\{tail(r) \mid B_r \in \mathbf{B}\}$
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^*.

An immediate goal aimed at in the near future is to improve the value of R^*.

4. Conclusion and perspectives

This chapter has approached the problem of optimizing the investment cost of modular machining lines (also called transfer lines) aimed at producing a family of products. The possibility of allowing variations over the set of products is the most important step for a manufacturing system to become reconfigurable. The specificity of the lines analyzed here is the parallelization of the operations' execution by the same spindle head. Due to the important investment cost required to build such lines, the search of an optimal design decision for the whole family appears as necessary. The potential economic benefits achieved are not negligible and is one of the motivating reasons to propose such an approach. The powerful mathematical programming tools make it possible to solve exactly and efficiently such problem, providing cost effective solutions. However, searching for the optimal solution may be prohibitively time-consuming, as much as the scale problem is larger.

The cost optimization of this kind of machining lines is a new and poorly studied problem, different from the classical SALB problem, but also NP-hard. This work presented a complete mathematical formulation of the problem as a linear program and proposed a procedure to follow for obtaining an exact (optimal) 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, to avoid running time exhaustion.

A particular attention should be focused on improving the bounds. Due to the exponential complexity of the integer linear programming solving algorithm, a bad behavior when increasing the problem's dimension is highly possible. The large number of constraints is a feature that will potentially allow the coupling of the presented exact method with different types of heuristics, able to provide good bounds to exact methods. We consider that, for applying the proposed method in real life environments, this coupling is definitely necessary.

5. References

Baybars, I. (1986). A survey of exact algorithms for the simple line balancing problem, *Management Science*, Vol. 32, pp. 909-932, ISSN 0025-1909

Becker, C. & Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing, *European Journal of Operational Research*, Vol. 168, No. 3, (1 February 2006), pp. 694-715, ISSN 0377-2217

Belmokhtar, S.; Dolgui, A.; Guschinsky, N.; Ihnatsenka, I. & Levin, G. (2004). 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.

Bratcu, A. I.; Dolgui, A. & Belmokhtar, S. (2005). Reconfigurable Transfer Lines Cost Optimization – A Linear Programming Approach, *Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation – ETFA 2005*, Lo Bello, L. & Sauter, T. (Eds.), pp. 625-632, ISBN 0-7803-9402-X, Catania, Italy, September 19-22 2005, IEEE, Piscataway, NJ, U.S.A.

Dashchenko A. I. (Ed) (2003). *Manufacturing Technologies for Machines of the Future 21st Century Technologies*, Springer.

Dolgui, A.; Guschinsky, N. & Levin, G. (1999). 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.), pp. 329-334, ISBN 0-7803-5670-5, Barcelona, Spain, October 18-20 1999, IEEE, Piscataway, NJ, U.S.A.

Dolgui, A.; Guschinsky, N. & Levin, G. (2000). *Approaches for transfer lines balancing*, Preprint No. 8, Institute of Engineering Cybernetics/University of Technology of Troyes

Dolgui, A.; Guschinsky, N.; Levin, G. & Harrath, Y. (2001b). 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*, P. P. Groumpos, A. P. Tzes (Eds.), pp. 36-41, ISBN 0080435548, Patras, Greece, July 12-14, 2000, Elsevier.

Dolgui, A.; Guschinsky, N.; Levin, G.; Louly, M. & Belmokhtar, S. (2004). 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)

Dolgui, A.; Finel, B.; Guschinski, N.; Levin, G. & Vernadat, F. (2005). A heuristic approach for transfer lines balancing. *Journal of Intelligent Manufacturing*, Vol. 16, No 2, pp. 159-171, ISSN 0956-5515

Dolgui, A.; Guschinsky, N. & Levin, G. (2006a). A special case of transfer lines balancing by graph approach. *European Journal of Operational Research* Vol. 168, No. 3, (1 February 2006), pp. 732-746, ISSN 0377-2217

Dolgui, A.; Finel, B.; Guschinski, N.; Levin, G. & Vernadat, F. (2006b). MIP Approach to Balancing Transfer Lines with Blocks of Parallel Operations, *IIE Transactions*, 2006, (In Press)

Erel, E. & Sarin, C. (1998). A survey of the assembly line balancing procedures, *Production Planning and Control*, Vol. 9, pp. 414-434, ISSN 0953-7287

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

Hitomi, K. (1996). *Manufacturing system engineering*, Taylor & Francis

Hutchinson, G. (1976). Production Capacity: CAM vs. transfer line, *IE*, September 1976, pp. 30-35

Johnson, J. R. (1988). Optimally balancing large assembly lines with FABLE, *Management Science*, Vol. 34, pp. 240-253, ISSN 0025-1909

Kamrani, A. K. & Logendran, R. (1998). *Group technology and cellular manufacturing: methodologies and applications*, Gordon and Breach

Koren, Y.; Heisel, U.; Jovane, F.; Moriwaki, T.; Pritschow, G.; Van Brussel, H. & Ulsoy, G. (1999). Reconfigurable Manufacturing Systems. *CIRP Annals*, Vol. 48, pp. 527-598, ISSN 0007-8506

Park, K.; Park, S. & Kim, W. (1996). A heuristic for an assembly line balancing problem with incompatibility, range and partial precedence constraints. *Computers and Industrial Engineering*, Vol. 32, No 2, pp. 321-332, ISSN 0360-8352

Pastor, R. & Corominas, A. (2000). Assembly line balancing with incompatibilities and bounded workstation loads. *Ricerca Operativa*, Vol. 30, pp. 23-45, ISSN 0390-8127

Rekiek, B.; Dolgui, A.; Dechambre, A. & Bratcu, A. (2002). State of art of assembly lines design optimization, *Annual Reviews in Control*, Vol. 26, No. 2, pp. 163-174, ISSN 1367-5788

Scholl, A. & Klein, R. (1998). Balancing assembly lines effectively: a computational comparison. *European Journal of Operational Research*, Vol. 114, pp. 51-60, ISSN 0377-2217

Scholl, A. (1999). *Balancing and sequencing assembly lines*, 2nd edition, Physica, Heidelberg

Son, S. Y., (2000). *Design Principles and Methodologies for Reconfigurable Machining Systems*, Ph.D. Dissertation, University of Michigan

Talbot, F. B.; Paterson, J. H. & Gehrlein, W. V. (1986). A comparative evaluation of heuristic line balancing techniques, *Management Science*, Vol. 32, pp. 430-454

Tang, L.; Yip-Hoi, D. M.; Wang, W. & Koren, Y. (2005a). Selection Principles on Manufacturing System for Part Family, *Proceedings of the 2005 CIRP International Conference on Reconfigurable Manufacturing* (CD-ROM)

Tang, L.; Yip-Hoi, D.; Wang, W. and Koren, Y. (2004). Concurrent line-balancing, equipment selection and throughput analysis for multi-part optimal line design. *The International Journal for Manufacturing Science & Production*, Vol. 6, Nos. 1-2, pp.71-81, ISSN 0793-6648

Youssef, A. Y. M. & ElMaraghy, H. A. (2005). A New Approach for RMS Configuration Selection, *Proceedings of the 2005 CIRP International Conference on Reconfigurable Manufacturing* (CD-ROM)

Zhang, G. W.; Zhang S. C. & Xu, Y. S. (2002). Research on flexible transfer line schematic design using hierarchical process planning. *Journal of Materials Processing Technology*, Vol. 129, pp. 629-633, ISSN 0924-0136

Manufacturing the Future

Edited by Vedran Kordic, Aleksandar Lazinica and Munir Merdan

ISBN 3-86611-198-3

Hard cover, 908 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, July, 2006

Published in print edition July, 2006

The primary goal of this book is to cover the state-of-the-art development and future directions in modern manufacturing systems. This interdisciplinary and comprehensive volume, consisting of 30 chapters, covers a survey of trends in distributed manufacturing, modern manufacturing equipment, product design process, rapid prototyping, quality assurance, from technological and organisational point of view and aspects of supply chain management.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

S. Belmokhtar, A.I. Bratcu and A. Dolgui (2006). Modular Machining Line Design and Reconfiguration: Some Optimization Methods, Manufacturing the Future, Vedran Kordic, Aleksandar Lazinica and Munir Merdan (Ed.), ISBN: 3-86611-198-3, InTech, Available from: http://www.intechopen.com/books/manufacturing_the_future/modular_machining_line_design_and_reconfiguration__some_optimization_methods

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen