Preprints of the 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, Russia, June 3 - 5, 2009

Hierarchical Scheduling of Mobile Robots in Production-Transportation Supply Chains

Eugene Levner*. Leonid Meyzin*.
Frank Werner**

**Holon Institute of Technology 52, Golomb St., Holon, 58102 Israel*
e-mail: {levner, meyzin}@hit.ac.il
***Otto-von-Guericke-Universität, Fakultät für Mathematik PSF 4120*
39016 Magdeburg

Abstract: In this paper we propose a decomposition approach that hierarchically integrates the *batching* and *local search heuristics* in manufacturing scheduling. The problem comprises two main interrelated stages embedded in any production-transportation supply chain, namely (i) scheduling processing of raw materials and robot's transportation operation within each individual cell, and (ii) scheduling of transportation and distribution of batches of semi-finished products between cells. Several efficient heuristic algorithms are proposed. This work has been motivated by a real-life problem of production planning for a CIM system for manufacturing of multi-component products served by robots.

Keywords: Efficient algorithms; Graph-theoretic models; Periodic movement; Polynomial models; Robot; Scheduling; Sieve algorithm; Local search.

1. INTRODUCTION

Modern flexible manufacturing systems are large scale systems. Enormous volumes of information data are required to describe, monitor and optimize them. Due to the large problem sizes, optimal planning and scheduling of the entire manufacturing system are impossible even using modern high-speed computers and advanced software. The best practical way to treat such a situation is finding suboptimal strategies for planning/scheduling based on a hierarchical decomposition of the manufacturing system, and, in turn, the hierarchical decomposition of the scheduling problems.

There are many time and space scales over which planning and scheduling decisions must be made. The longest-term decisions involve planning/scheduling of capital expenditure and strategic planning. The shortest-term decisions involve the schedules to load/unload individual parts and robot moves. The space scales include the machine level, cell level, shop level, factory level, etc. In practice, these decisions are made separately though they are dependent. In particular, each higher-level decision presents an assignment to the next-level decision-maker. In this paper, we will consider a production-transportation supply chain and study the corresponding scheduling problem arising at the machine-cell levels. Decisions will be made in a way that takes the entire problem data into account, and we consider decisions in two levels over space scales: machine and cell levels.

In the last decades, there has been considerable research on the scheduling of manufacturing systems grounded in the so-called *group technology* (*GT*) paradigm (Ham et al, 1985; Suresh and Kay, 1998; Kamrani and Logendran, 1998). The tenet of GT is to collect together a maximal set of similar jobs into a single family called a *batch* for production so as to minimize the setup operations and time for the jobs belonging to the same batch. On the other hand, in many manufacturing cells served by robots the robot moves and some set of operations (or activities) form the same pattern that is repeated cyclically an indefinite number of times, where the objective is to find an optimal order of the operations on available facilities (e.g., machines and transport devices) that optimizes a given performance criterion (which is usually related to process quality, system performance or customer satisfaction).

In this paper, we propose to take an approach that hierarchically integrates the *batching* and *cyclicity* (*periodicity*) factors in manufacturing scheduling. We pose the following two research questions: (i) How to fully exploit the *GT methodology* to minimize production setups in manufacturing? (ii) How to make better use of the *periodic (repetitive) characteristic* of the production, transportation and delivery processes? The problem comprises two main interrelated stages embedded in any production-transportation supply chain, namely (i) scheduling processing of raw materials and robot's transportation operation within each individual cell, and (ii) scheduling of transportation and distribution of batches of semi-finished products between cells. The first question is resolved by breaking all the parts onto batches so as each batch is repetitively processed by machines in an individual cell whereas the part batches are processed by cells. Our approach is to develop and study a two-stage *sieve-based* algorithm for solving the general cyclic batching scheduling problem that integrates the

batching and cyclicity (periodicity) factors at the two stages in one unifying framework. This work is motivated by a real-life problem of production planning for a CIM system consisting of multi-machine robotic cells for manufacturing of multi-component products which is up and running in the Laboratory of Robotics & Mechatronics at the Holon Technological Institute.

The paper is organized as follows. Section 2 presents the CIM system under consideration. In Section 3, we describe a graph-theoretic model for the problem, which permits its decomposition into well-solvable components. Section 4 presents a fast exact algorithm for solving an important special case whereas Section 5 describes efficient heuristic algorithms for the general problem. Section 6 concludes the paper.

2. DESCRIPTION OF A CIM SYSTEM

For each product, the technological process is determined in advance by the product engineer defining which elementary technological operations correspond to the processing of each part of the product, which tools and machines are to be used and in which order all technological operations are to be performed. We will consider a sufficiently general situation when manufacturing a product in a robotic cell involves a batch processing of its parts. In this case, we have to take into account the influence of group technology both on the expert system design and on a model of the problem itself. Thus, we define a batch as the subset of elementary operations that are related to a specific product and are processed together in a cell. In this sense, a robotic cell in our paper is considered as a single "batch processing machine", and each product may be presented as a batch of a network structure.

As is well known, the decomposition of the manufacturing system into subsystems may be implemented by combining mathematical programming techniques with knowledge-based expert rules. Batching decisions may be effectively implemented by a combination of integer programming modelware with expert rules. Notice that unlike hybrid batching-scheduling models studied by Kusiak and Bielli (1997), and Stecke (2005), in our scheduling problem there is no need in formal batching decisions since, for machine processing of complex products in a production line, the decomposition of the latter into robotic cells and clustering of parts into batches is unambiguously determined by a product designer and assumed to be introduced into the expert system in the form of informal expert rules known in advance. Once the techniques for a system decomposition and grouping of parts into batches are well known (see, for example, Kusiak and Bielli, 1997; Suresh and Kay, 1998; Selim et al., 1998; Cheng et al, 2001), the remaining problem is to schedule these batches (and, also, the parts within each batch) in tandem robotic cells so as to minimize the makespan (the completion time for the last batch processed).

The scheduling problem will be formalized as a generalization of the multi-machine Johnson problem and solved by combining a combinatorial algorithm with expert rules. The approach suggested may be considered as a further development of scheduling techniques for PERT-type projects with *time lags* and *overlapping,* previously investigated in the literature (see, for example, among others, Elmaghraby and Kamburowski 1992; Levner and Nemirovsky 1994; Levner et al. 1995a, 1995b, 2007; Crama et al. 2000).

We now provide a description of the production line which has motivated this work. The line consists of a number of machines in series. A set J of m products are to be produced in turn on the machines. Machines are grouped into cells, and parts of each product are grouped into disjoint families ("batches"). In this machine shop environment, we consider processing operations, and three more factors that are specific for the group technologies in flexible manufacturing and robotic systems, namely, the transportation operations, changeovers, and time lags. Let us consider these factors in more detail.

2.1. *Machine level*

The machine level scheduling includes the implementation of optimal robot moves and sequencing of machine operations. A robot arm movement can take seconds while a semiconductor oxidation operation can take hours. The issue at this level is the optimization of robot moves so as to maximize the throughput. Here, a large set of operations is grouped into tasks to be performed at stations along a production line. The objective is to minimize the cycle time of the periodic production process, which results in a maximum production rate.

Planning and scheduling of flexible manufacturing systems is based on group technology concepts and known to be consisting of three stages: decomposition of a manufacturing system into subsystems; clustering of orders into batches, and scheduling of the obtained batches on batch processing machines (Suresh and Kay, 1998; Al-Anzi et al., 2007). In today's computer-integrated manufacturing, these problems are not only very large (for example, hundreds of machines, and 10,000 parts) but they involve a number of constraints that makes the problem computationally intractable by standard mathematical programming tools. A much more adequate tool for solving these problems is the hierarchical decomposition (Ham et al., 1985; Kusiak and Bielli, 1997; Banaszak et al., 2000).

2.2. *Cell level*

At this time/space scale, one must consider the interactions of a small number of machines within a single cell. The important issues include routing and scheduling of robot moves between cells. Each cell is considered as a batch-machine. The problem is that of ensuring that the specified batch operations are actually produced. At this level, the detailed specifications of the batch operations are taken as given. The issue here is to move part batches between cells in a way that reduces unnecessary idle time of both machines and robots. The routing problem is to choose the sequence of robot moves and the scheduling problem is to choose the times at which the part batches visit the cells. The resources include machines, transportation elements (robots), and storage space.

2.3. *Transportation Operations*

We consider typical robotic cells in which transportation robots, a conveyor, and automated guided vehicles (AGV) transport batches of parts. Every batch is first moved in a pallet by an AGV from the centralized input storage to a cell, and then, after being processed in all the cells, the batch of finished parts (that is, a finished product) is moved in another pallet by another AGV to the output storage (see Fig. 1 where a fragment of a production line is depicted).

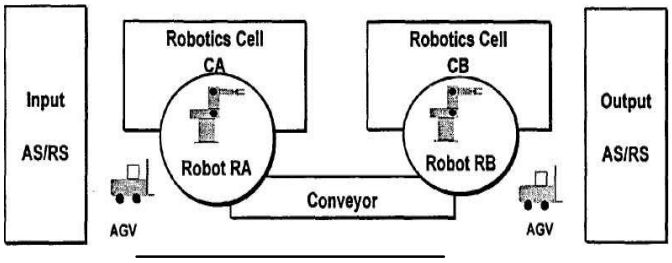

Fig. 1. Schematic of a production line

In this picture, after a current batch is delivered to the first cell, CA, from the input storage, robot RA participates in setting up the first machine in CA (for example, in changing tool magazines); then it waits until the part is finished at the machine; RA then participates in setting down the machine (including cleaning the machine and the adjacent area, and removing the used tools). Analogously, RA serves other machines in CA and then moves to its initial position where it starts its new life cycle with the next batch.

Similarly, robot RB starts its life cycle by participating in setting up the first machine of the cell CB, waits there until the part is finished on the machine and then participates in setting down the machine. Analogously, RB serves other machines in CB, and then returns a pallet filled with the finished parts to the output AGV. Then RB starts its next cycle by moving again to the first machine of CB.

Transportation of individual parts between the two cells is performed by a conveyor. Each part is moved from CA to CB immediately after finishing in CA, not waiting till all other parts of the batch are finished in the cell. The machine operations are synchronized in such a way that the transportation and processing of the next batch in cell CA can be started only after finishing all operations of the previous batch.

2.4. *Overchange Operations.*

In each cell, a major setup time is incurred when a machine changes from processing one part to another part that belongs to a different batch; along with that, a minor (low) setup time is incurred when a machine changes from processing one part to another of the same batch.

Whenever we change a machine to a new part or a new batch, the changeover time consists of two components, the "setup" time and the "setdown" time. We assume that the machines are not set up at time zero. In other words, processing of the first part in each cell must incur a major setup and then a minor setup. Similarly, the final setdown consists of the setdown of the last part plus the setdown of the last batch.

2.5. *Time Lags*

Associated with each batch j, $1 \leq j \leq m$, there are three arbitrary non-negative constants, S, T, and F, called *time lags*. It is assumed that batch j can start in cell CB only S time units after it has been started on CA; it may not be completed on CB sooner than T time units after its start on CA and F time units after its completion on CA. These time lags permit a realistic treatment of a variety of practical scheduling problems occurring in the robotic cell which cannot be treated by the standard Johnson problem. For example, a common practice is the *overlapping* of production - i.e., starting the processing of a batch in a cell CB after a certain minimum backlog of parts has been completed on CA; the backlog here may be easily represented by a time lag S. Similar examples for T and F time lags are evident. The time lags have another application as the auxiliary parameters used for a problem analysis and the design of optimal algorithms.

Now we are ready to formulate the objective of the problem. We have to find an order p of batches and, also, to schedule the parts within every batch which minimizes the maximum completion time under the following non-standard restrictions:

(i) Several machines are continuously available in each cell, capable to perform several operations at a time.
(ii) Each operation is performed only on one machine at a time.
(iii) Each batch being started is performed to its end with no part from any other batch intervening in the process. However, as soon as a part is finished in CA, it can be moved by the conveyor to the other cell, CB, not waiting till all other parts of the current batch are finished in CA.
(iv) There is an inter-machine storage space on the conveyor sufficiently large to store any number of parts finished in the cell CA.

We assume that the changeover and transportation times are not included into the processing times and appear as described explicitly above; all processing, changeover and transportation times are known constants, depending only on the batch being processed / transported, and not depending on the sequence in which batches and parts are processed and moved. We shall use the following notation:

m = number of products (batches),
n_j = number of parts in product j,
n = the total number of parts in all products,
k = the number of cells.

We assume that the group-technological process is described by a PERT network that presents technological operations, as well as their durations and precedence relations between them. Each manufacturing operation is performed by one or more cutting tools. Different cutting tools are stored in tool magazines, with automatic devices (for example, the feeding robots) performing their interchange. It is quite common for metal parts processed in robotic cells to require 10 to 30 different tools for their operation with 100-200 tools needed to produce different parts in a week. Therefore, significant tool switching delays usually accompany the processing

operations. Another source of delays to processing operations is the unreliability of the equipment and tools. The latter factor leads to time-consuming setup operations for serviceability control and on-line preventive maintenance of the equipment. The tool changeover and machine setup times are known to account to 20-25% of the total operation time. The setup times will significantly affect the quality of the machine schedules and the system performance. This makes scheduling of the setup operations to become a critical issue for the robotic cell management. Given the duration of all technological operations, tool/machine changeover and transportation times, the problem is to develop a good production plan over the planning cycle.

3. THE SCHDULING PROBLEM

The multi-machine production line which has motivated our work, has the last cell which is a "bottleneck" in the line (that is, the cell of minimal productivity), and a special buffer of a sufficiently large capacity for storing the batches arriving in turn, is placed before the cell. This fact allowed us to reduce the general problem to a relatively simple case of two robotic cells. We use the following heuristic techniques: all the cells before the bottleneck cell are aggregated into one, equivalent, cell denoted by CA, and the last cell is denoted by CB. We consider batch schedules in which all the batches are processed in the same order (such a schedule may be specified by a permutation of batches, and is called a *permutation schedule*). Hence, in what follows, instead of a multiple-cell scheduling problem, we consider a two-stage flowshop problem on two "batch processing machines" CA and CB. As we see below, this problem is a generalization of the classical two-machine Johnson problem when minimizing the makespan. Our approach is complementary to new heuristics suggested by Aldowaisan and Allahverdi (2004) for *m*-machine flowshop scheduling.

Consider an arbitrary permutation of batches $p = (1,2,..., m)$. Processing the batches in the sequence p can be graphically represented by an oriented graph $H(p)$, the nodes of which represent the technological operations and the arcs represent the precedence relations. The operation durations are assigned to the nodes, and called the *node weights*. Evidently, the network $H(p)$ consists of m (not necessarily identical) fragments, each of which corresponding to processing an individual batch j, $j = 1,..., m$.

For scheduling robotic cells, typically either the minimization of makespan or a traditional sum criterion such as the minimization of the weighted sum of completion time can be considered. In Section 4, we briefly describe how Johnson's algorithm for problem $F2||C_{max}$ can be generalized to the two-cell batch scheduling problem for minimizing the makespan. The two-cell batch scheduling problem is *NP*-hard in the strong sense for traditional sum criteria.

A recent overview on scheduling with setup times and costs has been given in Allahverdi et al., 2008. Current branch and bound algorithms can solve 2-machine batch scheduling problems with up to 30 – 35 jobs. An alternative is to formulated the resulting batch scheduling problems as a 0-1 mixed integer linear programming problem and to use a commercial software. However, due to the computational complexity of the model, only problems of rather small size (up to 20-25 jobs) can be solved exactly in this way. Since a typical practical problem includes often at least several dozen jobs, in Section 5 we describe several heuristic procedures that we have successfully applied to robotic scheduling.

4 GENERALIZED JOHNSON'S ALGORITHM

Assume that node Start denotes the beginning of the technological process in a given sequence p, and node Finish denotes the completion of the whole process. The arcs show only the precedence relations and do not represent real operations, and thus they have zero lengths. Denote by $d(p)$ an arbitrary directed path from node Start to node Finish in graph $H(p)$, and by $l(d(p))$, the length of $d(p)$, the latter being defined as the sum of the weights of all nodes in $d(p)$.

It is well known in the scheduling literature (see for example Johnson (1954)) that the makespan, $C_{max}(p)$, for any fixed permutation p equals the length $l_{max}(p)$ of the critical (the longest) path from Start to Finish in $H(p)$:

$$C_{max}(p) = l_{max}(p) = max_d\, l(d(p)) \qquad (1)$$

The following statement is valid:

Let p be an arbitrary sequence of m batches and of the parts within the batches, and let p^*_j be the optimal (i.e., the shortest) permutation schedule for a separate batch provided the latter is started at time 0. The makespan $C_{max}(p)$ does not increase if, for any j, $1 \le j \le m$, the parts in the batches are scheduled in the same sequence as in the schedule p^*_j.

The proof follows from (1) and the definition of makespan.

Define parameters A_j, B_j, C_j, as follows:

A_j = {*the length of the critical path in subgraph* $H_j(p)$ *from the node* **Start** $\mathbf{A_j}$ *to the node* **Finish** $\mathbf{A_j}$};
B_j = {*the length of the critical path in subgraph* $H_j(p)$ *from the node* **Start** $\mathbf{B_j}$ *to the node* **Finish** $\mathbf{B_j}$};
C_j = {*the length of the critical path in subgraph* $H_j(p)$ *from the node* **Start** $\mathbf{A_j}$ *to the node* Finish $\mathbf{B_j}$}.

We will use the following two-step batch scheduling algorithm providing an optimal solution for the two-cell batch scheduling problem, and solving approximately the original multi-cell problem. Its justification goes along the same line as that in Levner (1988) and Levner et al. (1995a).

Step 1. For each batch j, $1<j<m$, solve the multi-machine, n_j-*job* subproblem of minimizing the makespan in corresponding subnetwork H_j.

Denote by A_j, B_j and C_j the particular minimal makespan values for the subproblem corresponding to batch j.

Step 2. The order of parts in batches being determined according to the rule of Step 1, solve the 2-stage problem of scheduling m batches. For this purpose, first process batches for which $A_j \le B_j$, arranged so that the C_j-B_j values are ordered from smallest to largest, and then the remaining batches (i.e.,

such that $A_j > B_j$) arranged so that the $C_j - A_j$ values are ordered from largest to smallest.

5 FURTHER HEURISTIC ALGORITHMS

In this section, we describe several heuristic algorithms for robotic scheduling based on earlier algorithms by Sotskov et al. (1996) and Danneberg et al. (1999) developed for classical flow-shop batching problems with an arbitrary number of machines. The first paper considers the problem where the processing time of a batch is given by the sum of the processing times of parts (jobs) of the same group contained in one batch. For the case of makespan minimization, item availability is assumed, i.e., each part of a batch can be transported to the second machine immediately after the completion on the first machine. In Danneberg et al. (1999), jobs of the same group can be processed together in a batch but the maximal number of jobs in a batch is limited. In this case, the parts contained in a batch are processed in parallel, and the processing time of the batch is given by the maximal processing times of all parts contained in the batch. While the first type of problems is also denoted as sum-time batching problem, the second problem is referred to as a max-time batching problem.

For these types of batching problems, Sotskov et al. (1996) and Danneberg et al. (1999) describe several constructive and iterative heuristics which we adapt here for the robotic cell problem under consideration, where both types of problems may occur. Feeding and transportation robots perform or participate in setup operations on machines which occur before a batch is started.

We begin with the sum-time batching problem. The best constructive algorithm is an insertion algorithm combined with beam search. Insertion algorithms successively complete a partial permutation $p' = (p_1, p_2, ..., p_k)$ with $k < n$ to a complete sequence. Hence, in each step we have to determine which job has to be inserted next and on which position this job will be inserted.

For the makespan problem it has been found that an insertion of the jobs group by group according to non-decreasing sums of processing times can be recommended (i.e., first the job with the smallest sum of processing times is scheduled and then the remaining jobs of this group are inserted according to non-decreasing sums of processing times, then among the remaining jobs the job with the smallest sum of processing times is taken and so on). To apply beam search, a limited number of partial solutions is considered in parallel. If a beam width of b is used we select, at each step, b best partial sequences. Moreover, we suggest an iterative algorithm which is based on a complex shift neighbourhood. Namely, the shift neighborhood is applied in two levels. In the first level, shifts of batches are considered. This also includes splitting of a batch into two batches and shifting one of the new batches. A shift of a job within a batch is considered as a second level shift.

For the max-time batching problem, we have applied the problem specific modifications of the above algorithms for this type of problem. Now we assume batch availability of the jobs. Thus, the processing of a batch on the second machine can start if the processing of the preceding batch has been finished on this machine and a setup has been done and if the processing and transportation of the current batch to this machine has been completed. Now, the insertion algorithm turns into a 2-stage algorithm. In the first stage, batches of the jobs of the individual groups are determined by inserting successively jobs into the current batch. This is done by starting with the job having the smallest sum of processing times on both machines and including then further jobs into this batch such that the increase in the sum of the batch lengths on two machines is as small as possible in each case. After having determined the first batch of a group, one looks among the remaining jobs for one having the smallest sum of processing times and continue this procedure until all batches are formed.

The above procedure determines batches such that only the first batch may contain a smaller number of jobs that the maximal possible batch size and all other batches are completely filled. Then, in the second stage, the sequence of the batches is determined by applying an insertion algorithm to the batches that have been formed in the first stage. For this type of problems, it turned out that an insertion of the batches according to non-increasing sums of their lengths works best (this corresponds to the standard insertion algorithm for the flow shop problem without setup times).

The iterative algorithms use a similar neighborhood as it has been used for the sum-time batching problem. The only difference is that a batch may also be split without shifting a new sub-batch to another position (i.e., in the case of batch availability it might be advantageous to process two batches of the same group consecutively which is not considered for a sum-time batch problem with item availability). The generation of the different types of neighbors is controlled by probability parameters.

In addition to the standard metaheuristics, it turns out to be superior to apply multi-level heuristics (also denoted as iterated local search). The application of such heuristics to some scheduling problems has been described in Brucker et al. (1996) and Brucker et al. (1997). These algorithms use different neighborhood structures within the search. In particular, a neighborhood with large changes (high-level) and another neighborhood with small changes (low-level) are considered. Starting with an initial solution, these algorithms first perform a step in the high-level neighborhood and then a local optimization within the low-level neighborhood. After having determined a local optimum in the low-level neighborhood, the objective function value of this local optimum is compared with that of the initial solution using e.g. the acceptance criterion of simulated annealing. For these batching problems with makespan minimization, multi-level algorithms can be recommended where both the high-level and the low-level neigfhborhood are based on the complex shift neighborhood as well as a variant where the high-level neighborhood is based on

changes in the batch sequence and the low-level neighborhood is based job changes within two batches. For recommendations of the parameters used by the multi-level procedures, see Danneberg et al. (1999).

6. CONCLUSIONS

This paper has described a new approach for batch scheduling in a CIM system. The following advantages of the system have been displayed. First, the hierarchical two-level system approach has permitted us to formulate and simplify complex real-life batch scheduling problems with sophisticated logical and resource constraints that were aggregated into the Johnson-type problem of scheduling PERT projects on two machines. Second, the heuristics yielded approximate solutions better than existing FIFO-type heuristics on test and real-life data (see Danneberg et al. 1999).

Nevertheless, as the number of precedence relations in the networks grows, the running time of the suggested heuristic algorithms becomes prohibitively large, and the on-line re-scheduling becomes impossible. From this viewpoint, methods for accelerating the local search among the feasible solutions are to be further developed. Our further research will be devoted to the study and theoretical comparison of these methods as well as performing computational experiments for the comparison of the algorithm efficiency in practice.

REFERENCES

Al-Anzi, F.S., Allahverdi, A., and Kovalyov, M.Y. (2007), Batching deteriorating items with applications in computer communication and reverse logistics, *European Journal of Operational Research,* 182 (3) 1002-1011.

Aldowaisan, T., Allahverdi A. (2004), New heuristics for *m*-machine no-wait flowshop to minimize total completion time, *Omega*, 32 (5), 345-352.

Allahverdi, A., Ng, C.T., Cheng, T.C.E., and Kovalyov, M.Y. (2008), A survey of scheduling problems with setup times or costs, *European Journal of Operational Research*, 187, 985 – 1032.

Araujo, E.G., Dakin, F.A., Huber, M., Grupen, R.A. (1995), Hierarchical scheduling of robotic assembly operations in a flexible manufacturing system, *Int. J. of Flexible Automation and Inytegrated Manufacturing*, 3 (3-4), 301-216.

Banaszak, Z.A., Tang, X.Q., Wang, S.C., Zaremba, M.B. (2000) Logistics models in flexible manufacturing. *Computers in Industry*, 43, 237-248.

Brucker, P.; Hurink, J.; Werner, F. (1996). Improving local search heuristics for some scheduling problems – I, *Discrete Applied Mathematics*, 65, 97 – 122.

Brucker, P.; Hurink, J.; Werner, F. (1997). Improving local search heuristics for some scheduling problems. Part II, *Discrete Applied Mathematics*, 72, 47 – 69.

Cheng, C.H., Goh, C.-H. and Lee, A. (2001). Designing group technology manufacturing systems using heuristics branching rules, *Computers & Industrial Engineering*, 40 (1-2), 117-131

Crama, Y., Kats, V., Van de Klundert, J., and Levner, E., Cyclic scheduling in robotic flowshop, *Annals of operations Research,* 2000, 96(1-4), 97-123.

Danneberg, D.; Tautenhahn; T.; Werner, F. (1999). A comparison of heuristic algorithms for flow shop scheduling problems with setup times and limited batch size, *Mathematical and Computer Modelling*, 29, 101 – 126.

Elmaghraby, S.E. and Kamburowski, J. (1992). The analysis of activity networks under generalized precedence relations (GRP), *Management Science,* 38, 1245-1263.

Ham, I., Hitomi, K. and Yoshida T. (1985). *Group Technology - Applications to ProductionManagement,* Kluver-Nijhoff.

Johnson, S.M. (1954). Optimal two- and three-stage production scheduling with setup times included, *Naval Research Logistics Quarterly,* 1, 61-68.

Kamrani, A.K and Logendran, R. (1998) *Group Technology and Cellular Manufacturing: Methodologies and Applications,* Taylor & Francis.

Kusiak, A., M. Bielli (eds.) (1997) *Designing Innovations in Industrial Logistics Modelling (Mathematical Modeling)*, CRC Press, Boca Raton.

Levner, E.V. (1988). Efficient graph algorithms for solving Johnson-type scheduling problems. In: A.A. Fridman and E.V. Levner (eds.), *Economic-Mathematical Modeling and Analysis of Discrete Systems,* USSR Academy of Sciences, CEMI Press, Moscow,. 109-125.

Levner, E., Kats, V., and Alcaide, D. (2007). Cyclic Scheduling in Robotic Cells: An Extension of Basic Models in Machine Scheduling Theory. In E. Levner (ed.), *Multiprocessor Scheduling: Theory and Applications,* Itech Education and Publishing, Vienna, Austria.

Levner, E. , Kogan, K., and Levin, I. (1995a). Scheduling a two-machine robotic cell: A solvable case. *Annals of Operations Research,* 57, 217-232.

Levner, E., Levin, I., and Meyzin, L. (1995b). A tandem expert system for batch scheduling in a CIM system based on group technology concepts, *Proceedings of the IEEE/INRIA Symposium on Emerging Technologies and Factory Automation,* ETFA, Paris, 667-674.

Levner, E.V., and Nemirovsky, A.S. (1994). A network flow algorithm for just-in-time project scheduling, *European Journal of Operational Research,* 79(2)167-175.

Selim, H.M., Askin R.G. and Vakharia, A.J.(1998), Cell formation in group technology: Review, evaluation and directions for future research, *Computers & Industrial Engineering*, 34 (1), 3-20.

Sotskov, Y.; Tautenhahn, T.; Werner, F. (1996). Heuristics for permutation flow shop scheduling with batch setup times, *OR Spektrum,* 18, 67 – 80.

Stecke, K.E. (2005), Using mathematics to solve some problems in industry," INFORMS Transactions on Education, 5(2), 1-8.

Suresh, N.C., Kay, J.M. (eds) (1998). *Group technology and cellular manufacturing: a state-of-the-art synthesis of research and practice*, Springer.